Differential equations learning from spatial-time series data by the fast iterative shrinkage-thresholding algorithm

Pumipat Tongudom\textsuperscript{1}, Montri Maleewong\textsuperscript{1}
\textsuperscript{1}Department of Mathematics, Faculty of Science, Kasetsart University, Bangkok, Thailand

Email: pumipat.to@ku.th

Abstract. The numerical method of data-driven modeling is presented in this work. It can be used to find the corresponding governing equations in terms of differential equations from some provided spatial and time series data. The problem can be reduced to solve by sparse linear regression technique that can be referred to one of the powerful technique in machine learning algorithm. Here we propose to apply the Fast Iterative Shrinkage-Thresholding Algorithm (FISTA) and then study the capability of its predictions. The environmental problem involves advection and diffusion effects are demonstrated to show the accuracy and the efficiency for finding all of the coefficients in the partial differential equation corresponding to the input data.

1. Introduction
From the past decade until now, the data storage devices and the hardware have become much cheaper. The power of computing increases very fast. So, creating mathematical model has become more convenient and easier than before by using the collected data and efficient algorithms. In the past, differential equations are derived from the principles of physics or various conservative laws. These principles have been developed to describe natural phenomena. However, the problems in real situations such as neuroscience, environmental system, propagation of disease, or etc. have much complexities. It is very difficult to find underlying mathematical models to describe the physical phenomena.

In the present, there have been various numerical methods that can create differential equations from spatial and time series data, such as artificial neural network [1], nonlinear regression [2], sparse identification of nonlinear dynamical systems [3], and the PDE-FIND with the sparse linear regression [4]. The main objective of our present study is to apply the PDE-FIND to discover the governing equations from the collected data. The proposed PDE-FIND algorithm is modified using the FISTA algorithm which gives faster and better convergence rate than using the ISTA [5] and the TWIST [6]. In addition, we have demonstrated how to apply the proposed method for finding various parameters in the considering differential equations. This method can be applied to solve environmental problems involving advection and diffusion effects.

2. Theory and related works
2.1. Differential equation learning
In order to find the governing equations or mathematical models from the collected spatial and time series data. Generally, we assume that the differential equations are in the following form.
\[ u_t = N(u, v, u, u_x, u_{xx}, \ldots, x, t, \mu), \]  

(1)

where \( u_t \) is the derivative of an unknown function in time, \( N(\cdot) \) is an unknown operator, \( \mu \) is the parameter in our method. The main goal is to find \( N(\cdot) \) implying that the mathematical model can be found from the given data.

The right-hand side of (1) can be expressed in a vector form as

\[ U_t = \Theta(U, Q)\xi, \]  

(2)

where \( \Theta(U, Q) \) is the library of candidate terms in (1) that contains various derivative terms of \( u(x, t) \). \( Q \) represents any additional input terms, and \( \xi \) is the sparse vector identifying the derivative terms in the corresponding differential equations. Overall steps can be summarized in Figure 1. The challenging point is the optimization process to find numerically the sparse vector. The FISTA and ISTA methods are applied in this work. The final output will be the governing equations or PDEs that can represent the dynamics of collected data.

**Figure 1.** Overall steps in the differential equation learning for identifying mathematical model from the spatial and time series data.

### 2.2. Building libraries of candidate terms

The first step for building libraries of candidate terms starts from collecting all spatial and times series data, then rearrange them into a column vector \( U \in \mathbb{R}^{nm} \) of \( n \) spatial locations and \( m \) time data points. We collect \( U, Q \) in a library as

\[ \Theta(U, Q) \in \mathbb{R}^{nm \times d}, \]

where \( d \) is the number of candidate terms. The derivatives of all data with respect to time \( t \) and spatial \( x \) are calculated using the finite difference method. All of the candidate terms and the partial derivatives are represented in terms of \( \Theta(U, Q) \) as

\[ \Theta(U, Q) = [1 \ U \ U^2 \ \ldots \ Q \ \ldots \ U_x \ UU_x \ \ldots \ Q^2U^3U_{xx}]. \]  

(3)

Each column of \( \Theta(U, Q) \) contains candidate terms across of \( n \times m \) space-time grid points. For example, if we have 128 spatial data points, 100 time data points, and 20 candidate terms, then \( \Theta(U, Q) \in \mathbb{R}^{128 \times 100 \times 20} \).
2.3. Sparse Linear Regression

After we have already built the library and the corresponding candidate terms, we need to find the sparse vector $\xi$ for identifying the final candidate terms in the governing equation. In this work, the sparse linear regression method with the concept of the regularized least square can be applied as follows.

$$\xi = \arg \min \xi \left( U_l - \Theta \xi \right)^2 + \lambda \left\| \xi \right\|_1,$$

where $\lambda \left\| \xi \right\|_1$ is the penalty term, $\left\| \xi \right\|_1$ is the $l_1$-penalty term, $\lambda$ is a tuning parameter that controls the strength of the $l_1$-penalty and $\xi$ is the iterative sparse vector. More details can be found in [4].

2.3.1. Iterative shrinkage-thresholding algorithm (ISTA)

The iterative shrinkage-thresholding algorithm is one of the most powerful methods for solving equation (4) where each iteration involves matrix-vector multiplication involving $\Theta$ and $\Theta^T$ that are called the shrinkage or soft-threshold step. The general form of ISTA is that

$$\xi_{k+1} = \psi_a \left( \xi_k - t \Theta^T \left( \Theta \xi_k - U_l \right) \right),$$

where $t$ is a time step size, and $k$ is an iteration counter. The shrinkage operator $\psi_a$ is defined by

$$\psi_a \left( \xi_k \right) = \text{sign} \left( \xi_k \right) \left( \left| \xi_k \right| - \alpha \right)^+.\]$$

More details for calculating (5) and (6) can be found in [5, 8]. The advantage of ISTA method is that it is simple and accurate. However, the ISTA method may converge slowly for some problems. Short summary of the ISTA method is shown in algorithm 1.

2.3.2. Fast iterative shrinkage-thresholding algorithm (FISTA)

In some problems, the ISTA method may converge slowly. Then, in this work, we modify the PDE-FIND algorithm by applying the fast iterative shrinkage-thresholding algorithm (FISTA) in [7], and related work in [9]. The different concept between the ISTA and the FISTA is that the shrinkage operator $\psi_a \left( \xi \right)$ is not calculated directly, but rather for a vector $z_k$ which uses a linear combination of the first two previous vectors $\{\xi_{k-1}, \xi_{k-2}\}$. More additional computations are also performed using these formulae,

$$z_k = \xi_k, t_k = 1 \quad \text{and} \quad t_{k+1} = \frac{1 + \left( 1 + 4t_k \right)^{1/2}}{2}.$$\]

It can be proved from [7] that the convergence rates of the ISTA and the FISTA are $O \left( \frac{1}{k} \right)$ and $O \left( \frac{1}{k^2} \right)$, respectively, where $k$ is the iteration number. Short summary of the FISTA method is shown in algorithm 2.

Algorithm 1: ISTA($\Theta, U_l, \lambda, \text{iters}$)  
Algorithm 2: FISTA($\Theta, U_l, \lambda, \text{iters}$)
1. Set $\xi = [0]_{d\times1}$
2. Set a Lipschitz constant $L = \|\Theta^T\Theta\|$.
3. for $k = 1, \ldots, \text{iter}$:
   
   $\xi_{k+1} = \psi_u(\xi_k - t\Theta^T(\Theta\xi_k - U_t))$

4. If converge, return iterative sparse vector, else "error message"

The summary of the whole PDE-FIND method is shown in algorithm 3. The collected data in both spatial and time series data are divided into two sets composing of training and testing with the ratio of 80:20. The most important step in algorithm 3 is step 7 that solves the sparse linear regression problem. In this work, we will show the efficiency of the PDE-FIND algorithm by comparing the convergence rate between FISTA and ISTA methods.

**Algorithm 3:** TrainData($\Theta, U_t, \lambda, \text{max\_iters}, \text{iters}$) (modified from [3] and [4])

# Split the data into training and testing sets 80:20
1. Set $\Theta \rightarrow [\Theta_{\text{train}}, \Theta_{\text{test}}]$
2. Set $U_t \rightarrow [U_{t\text{\_train}}, U_{t\text{\_test}}]$

# Set an appropriate $l_0$ - penalty parameter.
3. Set $\lambda_0 = 10^{-3}$

# Get a baseline predictor
4. $\xi_{\text{best}} = (\Theta_{\text{train}})^{-1}U_{t\text{\_train}}$
5. $\text{error}_{\text{best}} = \|\Theta_{\text{test}} - \xi_{\text{best}}U_{t\text{\_test}}\|_2^2 + \lambda_0 \|\xi_{\text{best}}\|_0$

# Find the best predictor
6. for $i = 1, \ldots, \text{max\_iters}$:
   
   # Train and evaluate performance by FISTA
   6.1. $\xi = \text{FISTA}(\Theta_{\text{train}}, U_{t\text{\_train}}, \lambda, \text{iters})$
   6.2. $\text{error} = \|\Theta_{\text{test}} - \xi_{\text{best}}U_{t\text{\_test}}\|_2^2 + \lambda_0 \|\xi_{\text{best}}\|_0$
   6.3. If $\text{error} \leq \text{error}_{\text{best}}$:
      6.3.1. $\text{error} = \text{error}_{\text{best}}$
      6.3.2. $\xi_{\text{best}} = \xi$

7. Return $\xi_{\text{best}}$

3. Numerical experiment

To demonstrate the efficiency of the proposed differential equation learning method, we consider the environmental problem that involves advection, diffusion and reaction effects. The efficiency in terms
of the convergence rate are investigated. In this section, testing is done by using the numerical data from one-dimensional advection-diffusion problem. The problem is solved by the forward-time central space method (FTCS). The exact solutions are available in some cases. In order to measure the accuracy and the rate of convergence using the proposed algorithm, we define the mean absolute percentage error (MAPE) by

\[ \text{MAPE} = \frac{1}{n} \sum_{i=1}^{n} \left| \frac{\hat{w}_i - w_i}{w_i} \right| \times 100\% , \]

where \( n \) is the number of candidate terms, \( \hat{w}_i \) is the given exact coefficients in the PDE (if available), and \( w_i \) is the predicted numerical coefficients. For easy cases of boundary conditions, exact solutions can be found. It implies that the given data is exact without noise. But for the case of complex boundaries, the set of data are generated using the FTCS scheme. These numerical data can be regarded as data with noise. These two sets of data are used as input data in our method. Next, the accuracy can be investigated.

3.1. One-dimensional advection-diffusion problem
Consider the environmental problem involves advection and diffusion effects that can be represented by the PDE,

\[ \frac{\partial C}{\partial t} = D \frac{\partial^2 C}{\partial x^2} - u \frac{\partial C}{\partial x} , \]

where \( C(x,t) \) is the concentration of pollution, \( u \) is the flow velocity in the \( x \) direction, and \( D \) is the diffusion coefficient. We will consider two cases of Dirichlet and Neumann conditions.

3.1.1. Dirichlet boundary condition
The boundary and initial conditions are given by

\[ C(0,t) = 0 , \quad C(L,t) = 100 , \quad C(x,t) = \frac{100x}{L} , \quad 0 \leq x \leq L . \]

From [10], the exact solution is that

\[ C(x,t) = 100 \left[ \frac{p_x}{e^{\frac{P}{2}} - 1} + \frac{4\pi e^{2\pi}}{e^{\frac{P}{2}} - 1} \sum_{m=1}^{\infty} A_m + 2\pi e^{2\pi} \sum_{m=1}^{\infty} B_m \right] , \]

where \( P \) is the Peclet number defined by \( P = \frac{uL}{D} \), the coefficients \( A_m, B_m \) are given by

\[ A_m = (-1)^m \frac{m}{\beta_m} \sin \left( \frac{m\pi x}{L} \right) e^{-\lambda_m t} , \quad B_m = \left[ (-1)^m \frac{m}{\beta_m} \left( 1 + \frac{P}{2} \right) e^{\frac{P}{2}} + \frac{mP}{\beta_m} \right] \sin \left( \frac{m\pi x}{L} \right) e^{-\lambda_m t} , \]

where

\[ \beta_m = \left( \frac{P}{2} \right)^2 + (m\pi)^2 \quad \text{and} \quad \lambda_m = \frac{u^2}{4D} + \frac{m^2\pi^2D}{L^2} = \frac{D\beta_m}{L^2} . \]

The PDE is dimensionless form, we set \( u = -0.1 \), \( D = 0.01 \), and \( L = 2 \). The numerical data are generated by the FTCS scheme with \( \Delta x = 0.02 \), and \( \Delta t = 0.005 \). The plot of pollution concentration is shown in Figure 2. The numerical results are accurate. These two sets of data will be used as input data in algorithm 3. The problem is solved by FISTA and ISTA methods.
When the input data is numerical data, the PDE can be found as
\[ C_t = -0.10562C_x + 0.010114C_{xx}, \]
when the input data is exact with the same spacing in space and time, the PDE is that
\[ C_t = -0.100284C_x + 0.010118C_{xx}. \]

The number of iterations resulted from the ISTA and the FISTA algorithms versus mean absolute percentage error (MAPE) are shown in Figure 3. The convergence rate of FISTA is faster than ISTA. The MAPE versus the number of input data points is shown in Figure 4. When the number of input data increases, the present algorithm can find the PDE better.

### 3.1.2. Mixed boundary condition

The boundary conditions are given by
\[ C(0,t) = 20, \quad \frac{\partial C(L,t)}{\partial x} = 0, \quad t > 0. \]

To test the accuracy, the exact coefficients are given by \( u = -0.1 \), and \( D = 0.01 \). Here, we fix the pollution concentration at the left boundary while we leave naturally on downstream. Exact solution
can be found from [10]. Input data from the FTCS and analytical solutions provides respectively the PDE,
\[ C_t = -0.100041C_x + 0.009974C_{xx}, \quad \text{and} \quad C_t = -0.100048C_x + 0.01C_{xx}. \]
They are in good agreement when we use two different sets of input data (noise and without noise).

In term of convergence rate, FISTA method converges within 5 iterations while ISTA method converges within 10 iterations when the MAPE is set as 0.15%. Effects of increasing input data are shown in Table 1 where the number of spatial data points is fixed as 2000, then we vary the number of spatial data points. It shows that the errors decrease very fast when the number of data points increases.

**Table 1. Effects of increasing input spatial data points**

| Spatial points $n$ | PDE | MAPE(%) |
|--------------------|-----|---------|
| 10                 | $C_t = -0.0569C_x + 0.0046C_{xx}$ | 48.49 |
| 50                 | $C_t = -0.1004C_x + 0.0093C_{xx}$ | 3.50  |
| 200                | $C_t = -0.1001C_x + 0.0100C_{xx}$ | 0.17  |

4. Conclusions

The differential equation learning algorithm is presented in work. Input data will be separated as the training and testing sets, and then solve iteratively the sparse linear regression problem. Previously, the ISTA algorithm is implemented to find the coefficients in the PDE, but in this work, we have modified and implemented the FISTA algorithm in order to find all of the coefficients in the considering PDE. The accuracy and convergence rate are also shown. The testing PDE involves advection and diffusion effects with various types of boundary conditions. The PDE-FIND with FISTA method can be applied to find the target PDE which implies that all of the coefficients in the PDE can be estimated at the same time.

5. References

[1] González-García R, Rico-Martínez R and Kevrekidis I G 1998 Identification of distributed parameter systems: A neural net based approach *Computers chem. Engng* **22** 965
[2] Voss H, Kolodner P, Abel M and Kurths J 1999 Amplitude equations from spatiotemporal binary-fluid convection data *Phys. Rev. Lett.* **83** 3422
[3] Brunton S L, Proctor J L, and Kutz J N 2016 Discovering governing equations from data by sparse identification of nonlinear dynamical systems *Proc. Natl. Acad. Sci. USA.* **113** 3932
[4] Rudy S H, Brunton S L, Proctor J L, Kutz J N 2017 Data-driven discovery of partial differential equations *Science Advances* e1602614
[5] Daubechies I, Defrise M, and Mol C D 2004 An iterative thresholding algorithm for linear inverse Problems with a sparsity constraint *Comm. Pure Appl. Math.* **57** 1413
[6] Bioucas-Dias J and Figueiredo M 2007 Two-step iterative shrinkage-thresholding algorithms for image restoration *IEEE Trans. Image Process* **16** 2992
[7] Beck A and Teboulle M 2009 A fast iterative shrinkage-thresholding algorithm for linear inverse problems *Siam J. Imaging Science* **2** 183
[8] Tibshirani R 1996 Regression shrinkage and selection via the Lasso *J.R. Statist. Soc.* B **58** 267
[9] Nesterov Y E 1983 A method of solving a convex programming problem with convergence rate $O(1/k^2)$ *Dokl. Akad. Nauk SSSR* **27** 543
[10] Thongmoon M and Mckibbin R 2006 A comparison of some numerical methods for the advection-diffusion equation *Res. Lett. Inf. Math. Sci.* **10** 49