Physica A 556 (2020) 124839

Contents lists available at ScienceDirect

Physica A

journal homepage: www.elsevier.com/locate/physa

Exploring time-delay-based numerical differentiation using principal component analysis

Hongtao Li [a,b], Ersegun Deniz Gedikli [a,c,*], Raed Lubbad [a]

[a] Department of Civil and Environmental Engineering, NTNU, 7491 Trondheim, Norway
[b] DTU Space, Technical University of Denmark, 2800 Kgs. Lyngby, Denmark
[c] Department of Ocean and Resources Engineering, University of Hawaii at Manoa, Honolulu, USA

ARTICLE INFO

Article history:
Received 14 October 2019
Received in revised form 19 February 2020
Available online 18 June 2020

Keywords:
Time-delay-based differentiation
Principal component analysis
Embedding theory

ABSTRACT

Natural systems, including the dynamics of engineered structures, are often considered complex; hence, engineers employ different statistical methods to understand these systems better. Analyzing these systems usually require accurate derivative estimations for better understanding, i.e. a measured displacement can be used to estimate the forces on a cylindrical structure in water by using its velocity, and acceleration estimations. In this study, we use a nonlinear method based on embedding theory and consider the time-delay coordinates of a signal with a fixed lag time. We propose a new method for estimating the derivatives of the signal via redefining the delay matrix. That is, the original signal is updated with the second principal component of the delay matrix in each derivation. We apply this simple method to both linear and nonlinear systems and show that derivatives of both clean and/or noisy signals can be estimated with sufficient accuracy. By optimizing the required embedding dimension for the best derivative approximation, we find a constant value for the embedding dimension, which illustrates the simplicity of the proposed method. Lastly, we compare the method with some common differentiation techniques.

1. Introduction

Data collection and processing is a standard engineering practice used by industry and academia to understand the dynamics of a system in a given environment. These dynamics are usually complex; hence, engineers employ different statistical methods to understand the data better. For example, if the dynamical system is not known a priori and/or it is not possible to collect enough measurements to represent it, the time-delay coordinates of a single variable allow us to extract additional information (variables). The method of delays to study dynamics of experimental data was first proposed by Packard et al. [1]. Takens [2] improved the method and proved that the dynamics in the time-delay coordinates produce a new attractor with a similar topology. This is very important because it shows that it is possible to reconstruct the state–space from a low-dimensional time series, and when the dimension is high enough, reconstruction is always an embedding. For example, recently, Yu and Li [3] have used delay coordinates where the aircraft accidents for a certain time period have been reconstructed using chaos analysis.

In an earlier work, Broomhead and King [4] combined principal component analysis (PCA) with the method of delays and studied the effect of lag time on the reconstruction to eliminate linearly dependent coordinates and accurately

* Corresponding author at: Department of Civil and Environmental Engineering, NTNU, 7491 Trondheim, Norway.
E-mail addresses: deniz.gedikli@ntnu.no, egedikli@hawaii.edu (E.D. Gedikli).

https://doi.org/10.1016/j.physa.2020.124839

estimate the phase portrait, hence the underlying dynamics. They showed that when the time series has some Gaussian noise embedded in the signal, PCA is the optimal linear coordinate transformation and can reduce noise. Similarly, Shang et al. [5] developed a smoothing algorithm that is based on Chaotic Singular-Value Decomposition, which essentially relies on the use of delay coordinates and PCA. It should be noted that PCA is also known as the Karhunen–Loeve decomposition, proper orthogonal decomposition (POD) or singular-value decomposition (SVD), which has been widely used in all engineering areas to find coherent structures (or dominant modes), e.g., Kerschen et al. [6]; Gedikli et al. [7,8]. Therefore, we use PCA and SVD interchangeably throughout the paper.

Gibson et al. [9] studied PCA and the method of delays in depth and derived a closed-form solution to PCA in the limit of small window widths. They demonstrated that with the exception of the first principal component, the resulting principal components are qualitatively equivalent to average time derivatives. In addition, they identified the relationships between delays, derivatives, and principal component analysis and showed that eigenvectors of the system resemble Legendre polynomials. They derived a discrete set of Legendre polynomials and found stable estimates of derivatives using these polynomials.

In this paper, we investigate the method of delays combined with PCA to find reliable estimates of the derivatives. In this method, we create a Hankel matrix of a single variable (a single time-dependent measurement) and compute its principal components. We relate the second principal component of the data to the first derivative of the signal using an analytical formula. Then, instead of analytically estimating higher-order derivatives using Legendre polynomials, we treat the first time derivative of the signal as new signal. Therefore, we create a new Hankel matrix of the first derivative, compute the second principal component of the new matrix and estimate the second derivative. We also use the same approach to estimate the higher-order derivatives and find a constant value for the required embedding dimension that best approximates derivative by optimization. In addition, we show that our proposed method works better in estimating higher-order derivatives than Legendre polynomial approximations proposed by Gibson et al. [9] when the signal has a certain level of noise. Also, our new approach only requires one simple function as opposed to the use of different polynomial functions for different derivatives as in Gibson et al. [9]. Moreover, the new approach does not require tuning of any parameters to get reliable derivative estimations. Further, we show that we can handle the intrinsic inability of PCA to deal with traveling motions by mirroring the signal before applying our method.

We test the new method (with and without prior mirroring) by applying it to data from both numerical simulations (including 33 low-dimensional chaotic flow systems as in Sprott [10]) and experimental signals and compare the results. We show that the higher-order derivatives of both linear and nonlinear systems can be estimated with sufficient accuracy, and in some cases, the noise is reduced. For example, we apply the new method to analyze data from an experiment that was carried out in an ice tank. In the experiment, ice floes were marked with Qualisys markers and using motion tracking, the positions of the points were recorded. In addition, accelerometers were placed on some of the floes. Hence, the accuracy of the derivatives could be compared with direct measurements. When compared with the direct measurements (measured acceleration is compared with the estimated acceleration from displacement measurements), the proposed time-delay-based differentiation yields very close results, showing the accuracy of the method. To further increase the confidence in our proposed method, we compare results obtained by our method for signals collected in the ice tank with other 7 various differentiation methods. Comparison suggests that our method gives comparable/similar results with those from the other methods. In addition, the proposed method is applied on 17 electrocardiogram (ECG) experimental signals. Results show that our new method produces equivalent estimates with other common differentiators.

In Section 2, the new method of the time-delay-based differentiation or, in short, the TDD is described. In Section 3, the applications of the TDD method to different numerical and experimental systems are presented in more detail. These systems include the following: Case 1 is a quasi-periodic signal; Case 2 is a nonlinear mass–spring–dashpot system as in Chatterjee [11]; Case 3 is a nonlinear Chi-Chi earthquake system; Case 4 is a hysteretic Bouc–Wen system; and Cases 5–6 are experimental systems that consist of wave-induced ice floe collisions. In Section 4, the optimization results to find the embedding dimension and required shifting are presented. In Sections 5–7, we investigate the performance of the proposed TDD method and its variant on clean and noisy simulated signals and experimental signals, and illustrate the results of higher-order derivative estimations. In the end, summary and conclusions are provided in Section 8.

2. Time-delay-based differentiation (TDD)

TDD method is inspired by the idea of *state–space reconstruction using a time-delayed signal*. Previously, Gibson et al. [9] proposed the use of Legendre polynomials to estimate derivatives of the signals where one need to calculate different coefficients with increasing complexities for increasing order of derivatives. Although such a method gives good estimates of derivatives, it becomes highly time consuming as the required order of derivatives increase. In this study, we pose the following question: if we can create the state–space representation of a given signal using the combined SVD and method of time delays, can we also estimate the higher-order derivatives using this dynamic measurement considering the directional behavior of the signal? In other words, when the SVD of a Hankel matrix is considered, does the second principal component of the signal help us to estimate higher-order derivatives (i.e., second, third and so on)? If so, how can we relate the second principal component to higher-order derivatives?

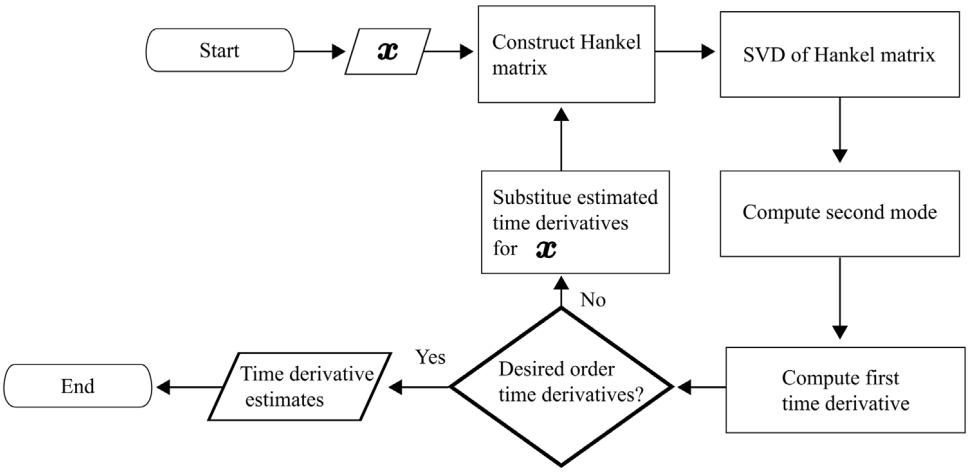

Fig. 1. Illustration of the TDD method.

Consider a signal with a time history of $\boldsymbol{x} \in \mathbb{R}^{n\times 1}$. We stack $\boldsymbol{x}$ to form a Hankel matrix $\boldsymbol{H} \in \mathbb{R}^{(n-r)\times r}$, which yields

$$\boldsymbol{H} = \begin{bmatrix} x_1 & x_2 & x_3 & \cdots & x_r \\ x_2 & x_3 & x_4 & \cdots & x_{r+1} \\ x_3 & x_4 & x_5 & \cdots & x_{r+2} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{n-r} & x_{n-r+1} & x_{n-r+2} & \cdots & x_{n-1} \end{bmatrix} \tag{1}$$

where the subscript represents the time instant when the entry of $\boldsymbol{x}$ is sampled, n is the length of the signal, and r is the embedding dimension and specifies the number of columns in $\boldsymbol{H}$ that are populated by the delayed $\boldsymbol{x}$.

$\boldsymbol{H}$ can be decomposed into three matrices by utilizing SVD:

$$\boldsymbol{H} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^{\mathrm{T}} \tag{2}$$

where columns of $\boldsymbol{U} \in \mathbb{R}^{(n-r)\times r}$ are the left orthonormal singular vectors of $\boldsymbol{H}$, columns of $\boldsymbol{V} \in \mathbb{R}^{r\times r}$ are the right orthonormal singular vectors of $\boldsymbol{H}$, and $\boldsymbol{S} \in \mathbb{R}^{r\times r}$ is the singular value matrix with its diagonal elements in descending order. In this decomposition, $\boldsymbol{U}$ includes temporal modes, and $\boldsymbol{V}$ contains spatial modes. In addition, only diagonal elements of $\boldsymbol{S}$, i.e., s_{ii}, contain units since column vectors of $\boldsymbol{U}$ and $\boldsymbol{V}$ are normalized.

In the following, we use $\boldsymbol{u}_i$ to denote the i^{th} column vector of $\boldsymbol{U}$ and v_{ii} to denote the i^{th} diagonal elements of $\boldsymbol{V}$ counted from the left upper corner. By using components of $\boldsymbol{U}$, $\boldsymbol{V}$ and $\boldsymbol{S}$, the i^{th} mode approximation of $\boldsymbol{x}$ can be computed as follows:

$$\boldsymbol{x}_i^d = \boldsymbol{u}_i s_{ii} v_{ii} \qquad i = 1,\ 2,\ \ldots,\ r \tag{3}$$

where superscript d means that the quantity is estimated by means of the method based on time delay. When $i = 1$, $\boldsymbol{x}_1^d$ is the reconstruction of $\boldsymbol{x}$.

Assuming the first time derivative of $\boldsymbol{x}$ can be best estimated with the second mode approximation, as in Eq. (3), we show that the first time derivative $\dot{\boldsymbol{x}}^d$ and the second time derivative $\ddot{\boldsymbol{x}}^d$ can be estimated as follows

$$\dot{\boldsymbol{x}}^d = \boldsymbol{x}^{(1),d} = \boldsymbol{x}_2^d F_s \tag{4}$$

$$\ddot{\boldsymbol{x}}^d = \boldsymbol{x}^{(2),d} = \boldsymbol{x}_3^d F_s^2 \tag{5}$$

where F_s is the sampling frequency, and $\boldsymbol{x}_2^d$ and $\boldsymbol{x}_3^d$ represent second and third modes, respectively. Although higher-order estimates of the derivatives can be approximated by higher subspace dimensions of SVD ($\boldsymbol{x}^{(k),d} = \boldsymbol{x}_{(k+1)}^d F_s^k$, where $\boldsymbol{x}^{(k)}$ is the k^{th} time derivative), error in the estimation increases significantly as the subspace dimension increases because most of the energy in the signal is embedded in the lower subspace dimensions. We call this estimation SVD-based differentiation to illustrate how it differs from the TDD method.

In the TDD method we ask: if SVD-based differentiation using the method of time delays, as shown above, produces reliable phase portraits (e.g., Gibson et al. [9], Broomhead and King [4]), can we repeat this process to estimate higher-order derivatives? For example, can we treat the first derivative of a signal as our new signal and estimate the second derivative; then, after estimating the second derivative, treat the second derivative as the new signal and estimate the third derivative and so on. If so, how accurate would the estimated derivatives be?

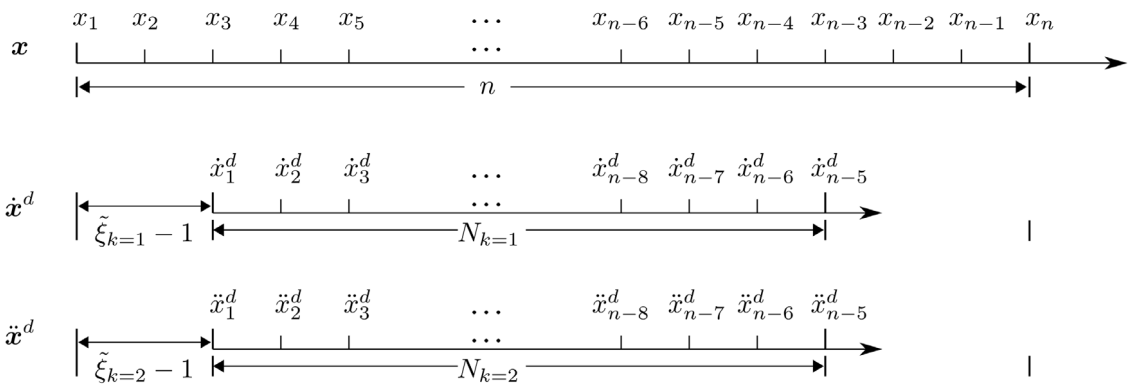

Fig. 2. Illustration of shifting for the SVD-based differentiation method when $r = 5$. $N_{k=1}$ and $N_{k=2}$ are the length of $\dot{\boldsymbol{x}}^d$ and $\ddot{\boldsymbol{x}}^d$, respectively.

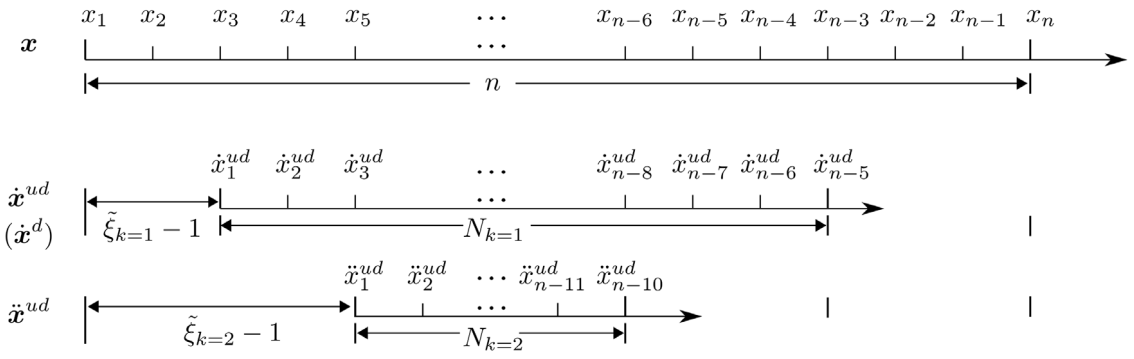

Fig. 3. Illustration of shifting for the TDD method when $r = 5$. $N_{k=1}$ and $N_{k=2}$ are the length of $\dot{\boldsymbol{x}}^{ud}$ and $\ddot{\boldsymbol{x}}^{ud}$, respectively.

In order to answer these questions, we propose the TDD method, which updates the original signal in the Hankel matrix with the estimated derivative and finds one higher-order derivative with the same approach (see Fig. 1 for the illustration of the TDD method). In other words, we only need Eq. (6) to estimate the second time derivatives in TDD:

$$\ddot{\boldsymbol{x}}_2^{ud} = \bar{\boldsymbol{x}}_2 F_s \tag{6}$$

where $\bar{\boldsymbol{x}}_2$ represents the second mode approximation using the updated Hankel matrix. It should be remembered that SVD-based differentiation and the TDD method yield the same results for the first derivatives since the first column of the Hankel matrix is the original signal in both methods. However, we update the Hankel matrix for each derivative in the proposed TDD method, whereas we directly use higher subspace dimensions to estimate the higher-order derivatives in SVD-based differentiation (or higher-order derivatives can also be estimated using SVD-based differentiation if they are combined with Legendre polynomials as in Gibson et al. [9]).

Takens [2] proved that if the dimension of the underlying dynamical system D_F is known, $r > 2D_F$ generally guarantees an embedding. In other words, when the dimension is sufficiently high, a reconstruction is almost always an embedding from the perspective of state–space reconstruction. There are many studies devoted to finding the minimum embedding dimension of a dynamical system for a reliable state–space reconstruction such as Kennel et al. [12], Cao [13], Kennel and Abarbanel [14], Maus and Sprott [15], Chelidze [16].

In our study, we use required embedding dimension that best approximates the derivatives, which is different from the minimum embedding dimension. Therefore, we create an optimization scheme to find the required embedding dimension that approximates the derivatives best (see Section 4).

Since our time-delay-based derivatives are shorter than the original signal, shifting of the signal is necessary to carry out the optimization. We show in Section 4 that the required embedding dimension ($\tilde{r}$) and the required shifting ($\tilde{\xi}_k - 1$) for the TDD method are 5 and $2k$, respectively, where k is the order of the time derivative. For the SVD-based differentiation, we show that $\tilde{r} = 5$ and the required shifting $(\tilde{\xi}_{k=1} - 1) = (\tilde{\xi}_{k=2} - 1) = 2$. Figs. 2 and 3 illustrate $\tilde{r}$ and $(\tilde{\xi}_k - 1)$ for the SVD-based differentiation and TDD method, respectively. As can be seen, the required shifting for the first two derivatives in the TDD method are $(\tilde{\xi}_{k=1} - 1) = 2$, $(\tilde{\xi}_{k=2} - 1) = 4$, respectively.

3. Applications of TDD in numerical and experimental systems

To test the performance of the TDD method, we use various numerical and experimental systems. First, we test the TDD method by applying it to a quasi-periodic signal (Case 1), and then, we apply it to various dynamical systems (Cases 2–6). Dynamical systems include numerical systems, such as the nonlinear mass–spring–dashpot system in Chatterjee [11] (Case 2), the nonlinear Chi-Chi earthquake system (Case 3) and the hysteretic Bouc–Wen system (Case 4), and experimental systems, such as the wave-induced ice floe collisions (Cases 5 and 6). Note that the six cases presented here are single-variable cases, i.e., each case describes a single variable. We also applied the TDD method to 33 different chaotic systems in Sprott [10] to test the performance of the method (not shown).

To illustrate the quality of the time derivatives, we compute the normalized root mean square error (NRMSE) as follows:

$$\mathrm{NRMSE} = \sqrt{\frac{\sum_{i=1}^{N_k} (a_{2k+i} - b_i)^2}{N_k}} (a_{max} - a_{min})^{-1} \tag{7}$$

In Eq. (7), a represents the analytical, numerical or measured time derivatives of the signal, and b represents the estimated derivatives using the TDD method. a_{max} and a_{min} are the maximum and minimum values of a, respectively. N_k is the length of the k^{th} time derivative (see Figs. 2 and 3).

3.1. Numerical systems

3.1.1. Case 1 — Quasi-periodic time series

We consider a quasi-periodic signal that is obtained by superimposing seven sinusoids with incommensurate frequencies. Eqs. (8), (9) and (10) show the analytical description of the signal with its corresponding first and second time derivatives, respectively. In this system, the sampling frequency is chosen as 50 Hz, and the signal is generated between 0 and 600 s.

$$\begin{aligned} x = {} & \sin(t) + \sin\left(\sqrt{2 + \frac{1}{\pi}}t\right) + \sin\left(\sqrt{5}t\right) \\ & + \cos\left(\frac{1}{3}t\right) + \cos\left(\frac{1}{7}t + 2\right) + \cos\left(\frac{1}{\sqrt{\pi}}t + \frac{\pi}{5}\right) + \cos\left(\sqrt{2}t + \frac{\pi}{9}\right) \end{aligned} \tag{8}$$

$$\begin{aligned} \dot{x} = {} & \cos(t) + \sqrt{2 + \frac{1}{\pi}} \cos\left(\sqrt{2 + \frac{1}{\pi}}t\right) + \sqrt{5}\cos\left(\sqrt{5}t\right) - \frac{1}{3}\sin\left(\frac{1}{3}t\right) \\ & - \frac{1}{7}\sin\left(\frac{1}{7}t + 2\right) - \frac{1}{\sqrt{\pi}}\sin\left(\frac{1}{\sqrt{\pi}}t + \frac{\pi}{5}\right) - \sqrt{2}\sin\left(\sqrt{2}t + \frac{\pi}{9}\right) \end{aligned} \tag{9}$$

$$\begin{aligned} \ddot{x} = {} & -\sin(t) - \left(2 + \frac{1}{\pi}\right)\sin\left(\sqrt{2 + \frac{1}{\pi}}t\right) - 5\sin\left(\sqrt{5}t\right) - \frac{1}{9}\cos\left(\frac{1}{3}t\right) \\ & - \frac{1}{49}\cos\left(\frac{1}{7}t + 2\right) - \frac{1}{\pi}\cos\left(\frac{1}{\sqrt{\pi}}t + \frac{\pi}{5}\right) - 2\cos\left(\sqrt{2}t + \frac{\pi}{9}\right) \end{aligned} \tag{10}$$

Fig. 4 shows the original signal with its estimated derivatives and the corresponding phase plane plots of second time derivative versus first time derivative using both SVD-based differentiation and the TDD methods. As seen, SVD-based differentiation fails to estimate the second time derivative of the signal (see Fig. 4c), obtaining an NRMSE value of 14.9%; on the other hand, TDD successfully estimates the second time derivative, achieving an NRMSE value of 0.0446% (see Fig. 4d). One can also compare the derivatives by observing their phase plane plots. In this case, Fig. 4e also shows that SVD-based derivative estimation fails to reconstruct the phase plane, whereas TDD-based derivative estimation (Fig. 4f) successfully reconstructs the desired phase plane.

3.1.2. Case 2 - A simple one-dimensional nonlinear mass–spring-dashpot system

We utilize the nonlinear mass–spring-dashpot system presented in Chatterjee [11] to examine the applicability of SVD-based differentiation and TDD for a nonlinear system. Fig. 5 shows the sketch of this nonlinear system. Each mass from m_1 to m_{10} is subjected to a force from a linear spring with stiffness k, a force from a linear dashpot with a constant coefficient of c and an external forcing function $F(t)$. Additionally, a nonlinear force from a hard spring k_h is exerted on mass m_5. This mass–spring-dashpot system moves on frictionless surface.

The equations of motion for this system can be written as follows:

$$m\ddot{x}_1 + 2c\dot{x}_1 + 2k_s x_1 = F(t) + k_s x_2 + c\dot{x}_2 \tag{11}$$

$$m\ddot{x}_i + 2c\dot{x}_i + 2k_s x_i = F(t) + k_s\left(x_{i+1} + x_{i-1}\right) + c\left(\dot{x}_{i+1} + \dot{x}_{i-1}\right) \tag{12}$$

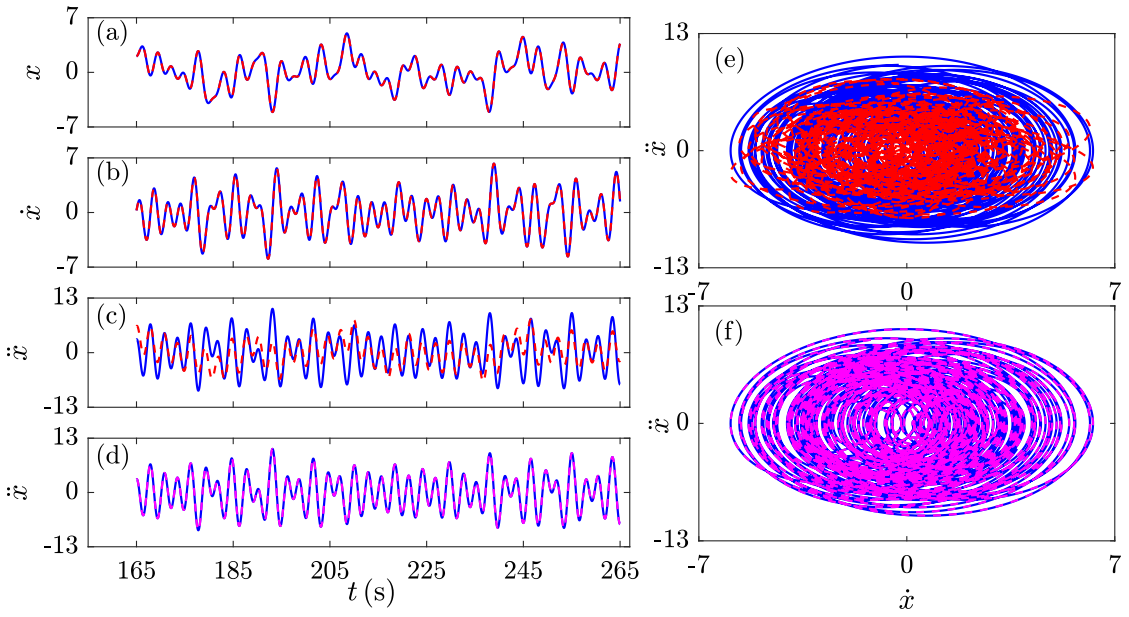

Fig. 4. Case 1. (a) represents the time history of quasi-periodic signal x. (b) is the first time derivative $\dot{x}$. (c)–(d) illustrate the second time derivative $\ddot{x}$ using SVD-based differentiation and TDD, respectively. (e)–(f) illustrate phase-plane plots using SVD-based differentiation and TDD, respectively. In the figures, the blue line represents analytical results, the red dashed line represents the results obtained using SVD-based differentiation, and the magenta dashed line represents the results obtained using TDD. Phase planes are plotted for $0 \sim 270$ s.

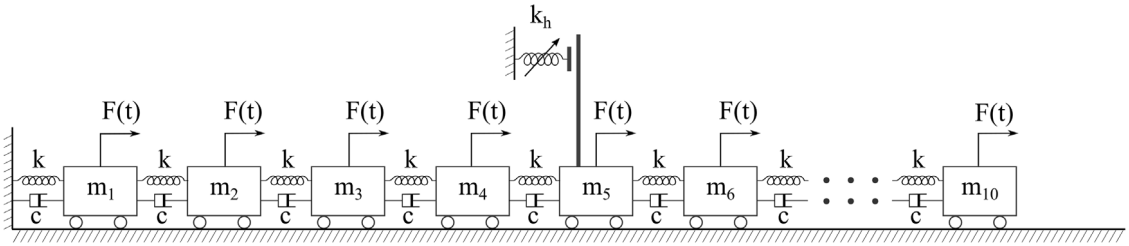

Fig. 5. Case 2. Schematic of the mass–spring-dashpot system with a nonlinear spring.
Source: Adapted from [11].

$$m\ddot{x}_5 + 2c\dot{x}_5 + 2k_s x_5 + \mathrm{H}(-x_5)k_h x_5^3 = F(t) + k_s\,(x_6 + x_4) + c\,(\dot{x}_6 + \dot{x}_4) \tag{13}$$

$$m\ddot{x}_{10} + c\dot{x}_{10} + k_s x_{10} = F(t) + k_s x_9 + c\dot{x}_9 \tag{14}$$

where $\mathrm{H}(x)$ is the Heaviside function, the subscript represents different masses, i stands for mass $2, 3, \ldots, 9$ excluding 5, and $F(t) = A\sin\omega t$ is the external force. Here, $m_1 = m_2 = \cdots = m_{10} = 1$, $k_s = 1$, $c = 0.3$, $A = 0.08$, $\omega = 0.2$ and $k_h = 5$, which are the same nondimensional values as in Chatterjee [11]. The sampling rate is chosen as 100 Hz, and the simulation time is $0 \sim 600$ s.

Fig. 6 shows the original displacement time history of the simple spring–mass–dashpot system with its estimated derivatives and the corresponding 3-dimensional phase plane plots using both SVD-based differentiation and TDD methods. Similar to the previous case, SVD-based differentiation once again fails to estimate the second time derivative of the signal (see Fig. 6c), obtaining an NRMSE value of 3.43%; on the other hand, TDD successfully estimates the second time derivative, with the NRMSE value of 0.0949% (see Fig. 6d). One can also compare the derivatives by observing their phase plane plots. In this example, Fig. 6e also shows that SVD-based derivative estimation does not reconstruct the phase plane accurately, whereas TDD-based derivative estimation (Fig. 6f) successfully reconstructs the desired phase plane with a high accuracy.

3.1.3. Case 3 — Nonlinear Chi-Chi earthquake system

In this case, we consider a single degree of freedom linear system with a nonlinear force, as presented in Smyth and Wu [17]. The system represents the Chi-Chi earthquake and can be expressed as follows:

$$m\ddot{x} + c\dot{x} + k_s x = -m\ddot{x}_g \tag{15}$$

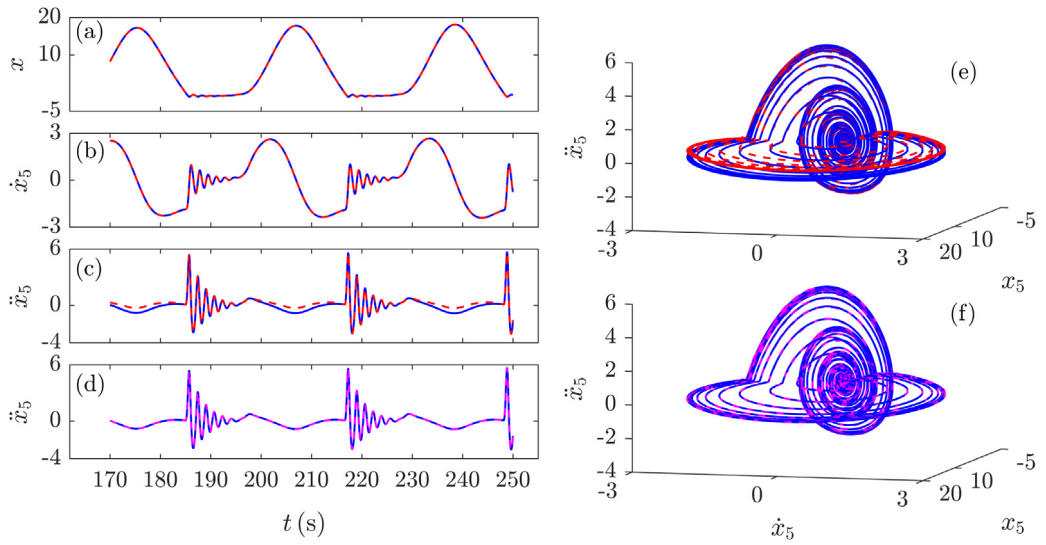

Fig. 6. Case 2. (a) represents the time history of the displacement x_5. (b) represents the time history of the velocity $\dot{x}_5$. (c)–(d) illustrate the acceleration time history $\ddot{x}_5$ of mass 5 using SVD-based differentiation and TDD, respectively. (e)–(f) illustrate 3-dimensional phase-plane plots using SVD-based differentiation and TDD, respectively. In the figures, the blue line represents simulation results, the red dashed line represents the results obtained using SVD-based differentiation, and the magenta dashed line represents the results obtained using TDD. Phase planes are plotted for 0 $\sim$ 600 s.

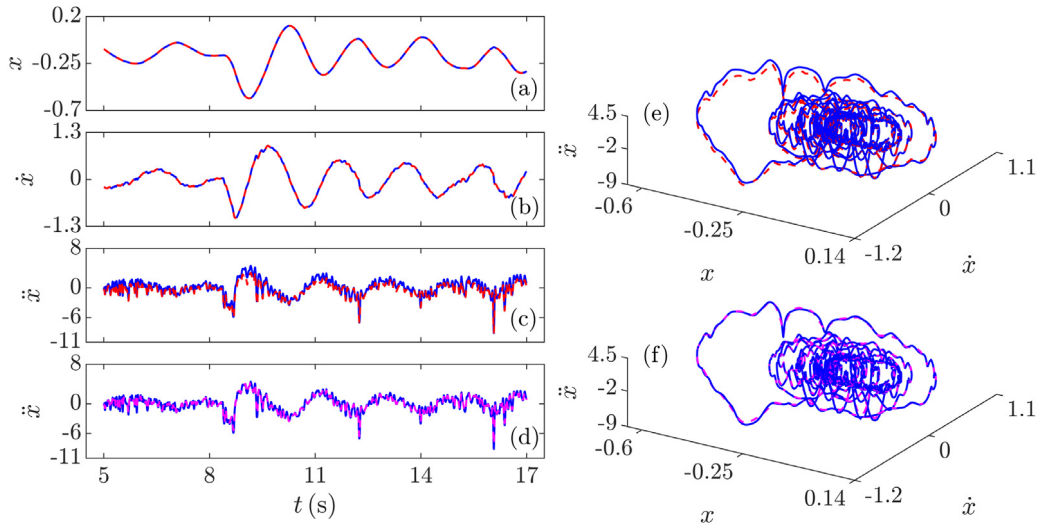

Fig. 7. Case 3. (a) represents the time history of the displacement x. (b) represents the time history of the velocity $\dot{x}$. (c)–(d) illustrate the acceleration time history $\ddot{x}$ using SVD-based differentiation and TDD, respectively. (e)–(f) illustrate 3-dimensional phase-plane plots using SVD-based differentiation and TDD, respectively. In the figures, the blue line represents simulation results, the red dashed line represents the results obtained using SVD-based differentiation, and the magenta dashed line represents the results obtained using TDD. Phase planes are plotted for 0 $\sim$ 25 s.

where $m = 1$, $c = 0.3$, and $k_s = 9$ and $\ddot{x}_g$ represents the ground acceleration. The system is excited by ground accelerations measured at 200 Hz at Taichung, Taiwan, during the Chi-Chi earthquake [17]. The data are provided by the Seismology Center, Central Weather Bureau in Taipei, Taiwan, through the strong motion virtual data center [18].

As in Smyth and Wu [17], velocity and displacement are obtained by using the Newmark method. Thereafter, Eq. (15) is solved to find acceleration. Simulation is performed for 0 $\sim$ 160 s.

Fig. 7 shows the original displacement time history of the nonlinear Chi-Chi earthquake system with its estimated derivatives and the corresponding 3-dimensional phase plane plots using both SVD-based differentiation and TDD methods. As clearly seen, SVD-based differentiation fails to estimate the second time derivative of the signal (see Fig. 7c&e), obtaining the NRMSE value of 3.45% and underestimating the nonlinearity in the signal; on the other hand, TDD performs

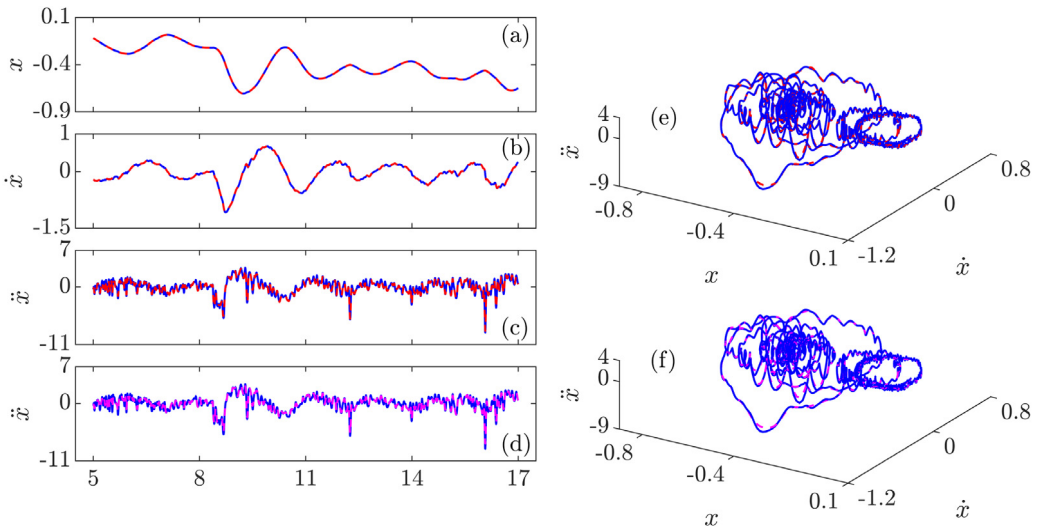

Fig. 8. Case 4. (a) represents the time history of the displacement of x. (b) represents the time history of the velocity of $\dot{x}$. (c)–(d) illustrate the acceleration time histories of $\ddot{x}$ using SVD-based differentiation and TDD, respectively. (e)–(f) illustrate 3-dimensional phase-plane plots using SVD-based differentiation and TDD, respectively. In the figures, the blue line represents simulation results, the red dashed line represents the results obtained using SVD-based differentiation, and the magenta dashed line represents the results obtained using TDD. Phase planes are plotted for $0 \sim 25$ s.

better at estimating the second time derivative, with the NRMSE value of 1.24% (see Fig. 7d&f), where the inherent nonlinearity in the signal is correctly estimated.

3.1.4. Case 4 — Single degree of freedom hysteretic Bouc–Wen system

The single degree of freedom hysteretic Bouc–Wen system that includes geometric nonlinearity is considered here, driven by the same earthquake data as in the previous case. The governing equations for this system are as follows:

$$m\ddot{x} + c\dot{x} + k_s\eta = -m\ddot{x}_g \tag{16}$$

$$\dot{\eta} = \dot{x} - \beta|\dot{x}||\eta|^{\alpha-1}\eta - \gamma\dot{x}|\eta|^{\alpha} \tag{17}$$

where η is the Bouc–Wen hysteretic component, $m = 1$, $c = 0.3$, $k_s = 9$, $\beta = 2$, $\gamma = 1$, and $\alpha = 2$. For more information regarding this system, readers are also encouraged to read Smyth and Wu [17]. The simulation is conducted for $0 \sim 160$ s, as in Case 3.

The displacement, velocity and acceleration are found first by utilizing the MATLAB built-in function *ode45* to solve the dynamic system equations, Eqs. (16) $\sim$ (17). Thereafter, the SVD-based differentiation and TDD methods are applied, and the results are compared as shown in Fig. 8. It is evident that both the SVD-based differentiation and TDD methods estimate the first time derivatives quite well (see Fig. 8b). The acceleration estimations using SVD-based differentiation and TDD are very close to the simulated results, as shown in Fig. 8 (c, d, e and f). The quality of the TDD estimation of acceleration is similar to that of SVD-based differentiation when comparing the NRMSE values, with an NRMSE in SVD-based differentiation of 0.963% and an NRMSE in TDD of 1.23%.

3.2. Experimental systems

In this section, the quality of the TDD method in estimating derivatives is investigated by examining two different dynamical systems from experiments. The experimental data utilized here were collected during two experimental test runs (i.e., test runs #3210 and #3220, respectively) of the LS-WICE project (see Tsarau [19], Cheng et al. [20], Herman et al. [21], Tsarau et al. [22] and Li and Lubbad [23] for details).

Fig. 9 shows the sketch of the experimental setup, where the wavemaker is capable of generating waves with different wave frequencies, wave amplitudes and wavelengths (hence various wave steepnesses). In the experiments, 28 (longitudinally) $\times$ 6 (transversely) rectangular ice floes with dimension 1.5 m by 1.63 m were produced in total by cutting a 36 mm thick intact ice sheet. In the tests, ice floes were instrumented with Qualisys markers, and an optical Qualisys system monitored 3-dimensional spatial positions of 5 ice floes with a frame rate 100 *frames/s*. To study the dynamics of wave-induced ice floe collisions, two inertia measurement units (IMUs) were deployed on two ice floes that provided direct acceleration measurements. Regular waves were generated by the wavemaker and propagated into the ice field. Two ultrasound sensors were mounted before the leading edge of the ice field to record the produced wave profiles. In the tests, ice floes moved predominately in the vertical plane under unidirectional monochromatic waves.

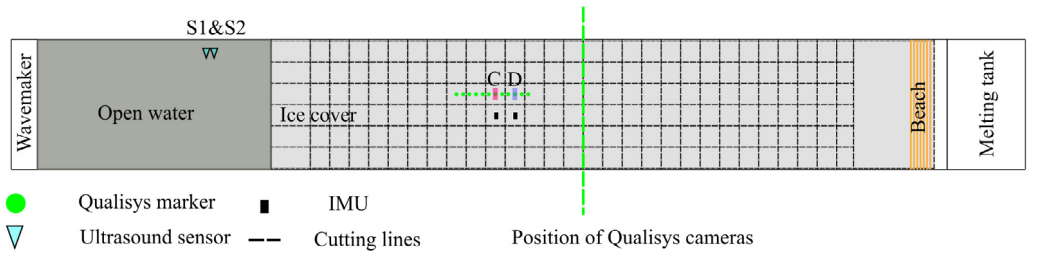

Fig. 9. Simplified sketch of the experimental setup, where S1 and S2 are ultrasound sonars to measure wave elevations. The flow is generated by the wavemaker and flows to the right. Two markers of interest are highlighted by magenta and light blue color. The ice floes where these two markers were placed on are assigned with letters C and D, respectively.

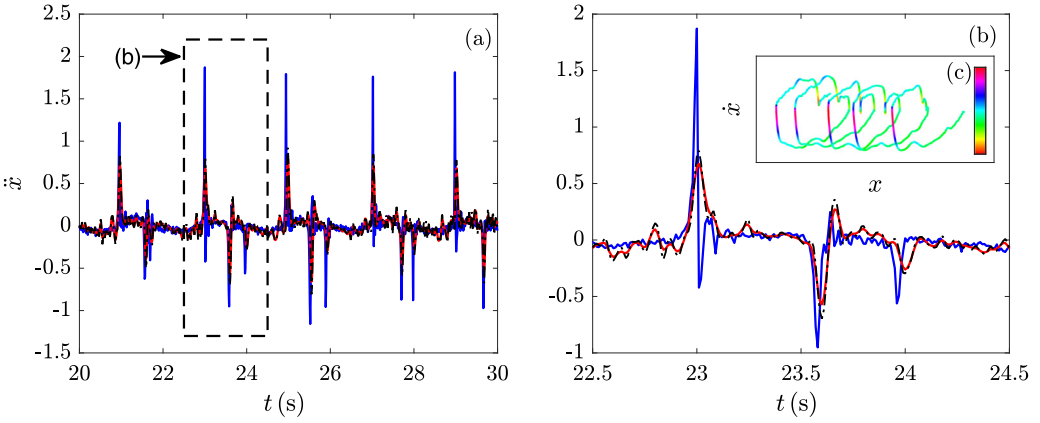

Fig. 10. Case 5 (wave steepness is 1.27×10^{-2}). (a) Comparison between acceleration measurements and estimates, where the blue line represents the measured acceleration obtained from the IMU, and the red and black lines represent the estimated accelerations from the TDD method and the DF, respectively. (b) is the magnified view of the part indicated by the black box in (a). (c) shows the phase-plane plots for $20 \sim 30$ s using the TDD method. In (c), the magenta color indicates the maximum positive acceleration, whereas the red color represents maximum negative acceleration.

In this subsection, the position of the selected marker on ice floe C is analyzed for test runs #3210 and #3220 (wave steepnesses are 1.27×10^{-2} and 1.56×10^{-2}, respectively), where the selected marker on ice floe C is highlighted by magenta color in Fig. 9. The reason we selected two different runs is because these two different runs caused different dynamical motions of the ice floes such that different nonlinearities were observed (see Figs. 10c and 11c).

As shown in Fig. 9, the velocity and acceleration of the ice floe under consideration were not measured directly; hence, a differentiator filter [24, referred as DF henceforth] and the TDD method were both applied to estimate them. It should be remembered that IMUs and markers were placed on neighboring ice floes, and videos of the experiment showed that the two neighboring ice floes moved together; therefore, it is expected that similar accelerations will be obtained when the motion of the two ice floes is compared (see more details in Li et al. [25]).

3.2.1. Case 5 — Wave-induced ice floe collisions (wave steepness is 1.27×10^{-2})

In this case, we apply the TDD method to wave-induced ice floe collisions, where the wave steepness is 1.27×10^{-2}. Fig. 10 shows the measured (blue) and estimated accelerations using the TDD method (red) and the DF (black), respectively, where we exclude the results from SVD-based differentiation method for clarity. As a result, the resulting acceleration estimations using both the TDD and DF successfully generate the acceleration signals as expected. However, it should be pointed out that the amplitudes of the resulting estimations are not the same as the direct measurements. This is attributed to the interaction between the ice floes (see Fig. 9) and the weight of the IMUs since each IMU is very heavy compared to the weight of the ice floe (1.5% of the ice floe weight); therefore, a large inertia is produced.

3.2.2. Case 6 — Wave-induced ice floe collisions (wave steepness is 1.56×10^{-2})

In this case, we apply the TDD method to the wave-induced floe collisions when the wave steepness is 1.56×10^{-2}. Similar to the previous case, Fig. 11 shows that the computed accelerations using both the TDD method and the DF yield similar results. However, one can also observe strong nonlinear interactions in the phase portrait in Fig. 11c, which differs from the system in the previous case in Fig. 10c.

For application of TDD method on more experimental signals, reader is referred to Section 7.

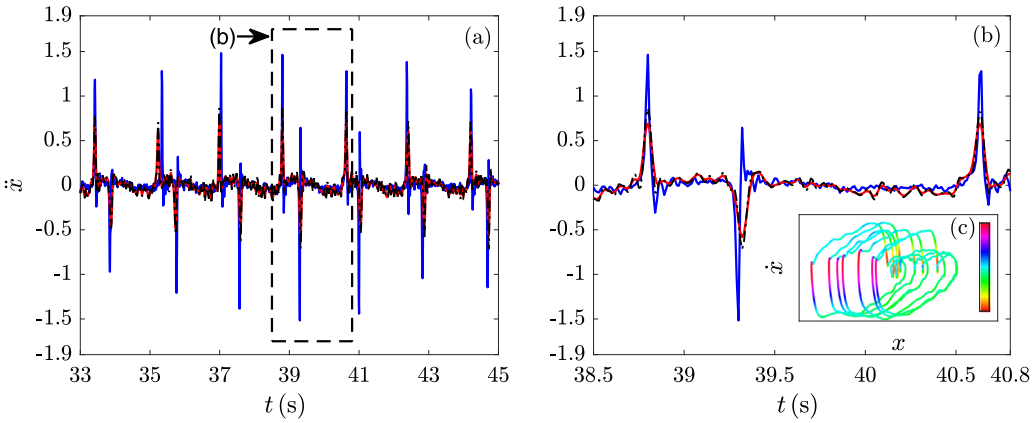

Fig. 11. Case 6 (wave steepness is 1.56×10^{-2}). (a) Comparison between acceleration measurements and estimates, where the blue line represents the measured acceleration obtained from the IMU, and the red and black lines represent the estimated accelerations from the TDD method and the DF, respectively. (b) is the magnified view of the part indicated by the black box in (a). (c) shows the phase-plane plots for 33 $\sim$ 45 s using TDD. In (c), the magenta color indicates the maximum positive acceleration, whereas the red color represents the maximum negative acceleration.

4. On the required embedding dimension and required shifting

Both SVD-based differentiation and the TDD methods rely on the critical parameter, viz., embedding dimension. The embedding dimension used in this work differs from the notion of minimum embedding dimensions (e.g., Kennel et al. [12], Cao [13], Kennel and Abarbanel [14], Maus and Sprott [15], Chelidze [16]), where we optimize the embedding dimensions to find the best derivative approximations when compared with the analytical/simulated results. The optimization problem can be solved by minimizing the following function:

$$\varepsilon_k(r, \xi_k) = \frac{1}{N \times N_k} \sum_{l=1}^{N} \sum_{i=1,\, j=\xi_k}^{i=N_k,\, j=N_k+\xi_k-1} |x_{l,i}^{(k),d} - x_{l,j}^{(k),s,t}| \tag{18}$$

$$\xi_k = 1,\ 2,\ \ldots,\ \tau_k \tag{19}$$

$$\tau_k = \begin{cases} \sum_{q=1}^{k-1} \tilde{r}_q + r & \text{for the TDD method} \\ r & \text{for the SVD-based differentiation method and } k = 1 \end{cases} \tag{20}$$

$$N_k = n - \tau_k \tag{21}$$

where $\boldsymbol{x}_l$ and N are the time series of the l^{th} variable and the number of variables in a signal system of interest, respectively; N_k is the length of the k^{th} time derivatives; subscripts i and j are the indices of entries (used for fidelity assessment) in the data vectors of $\boldsymbol{x}_l^{(k),d}$ and $\boldsymbol{x}_l^{(k),s,t}$, respectively; superscripts s and t denote simulated and theoretical, respectively; $\tilde{r}$ is the required embedding dimension that best approximates the time derivatives; τ_k is the truncation for the k^{th} time derivatives relative to the signal because of the time delay; n is the length of the signal; and r is the embedding dimension tested, ranging from 3 to 20. Note that the SVD-based differentiation method requires the same embedding dimension and shifting for all orders of time derivatives. Thus, we only perform optimization for $k = 1$.

According to Eqs. (18) $\sim$ (21), when $\varepsilon_k(r, \xi_k)$ reaches the minimum for a signal with different embedding dimensions r and different shiftings $(\xi_k - 1)$, it results in the best derivative approximation.

Optimization is applied to 33 low-dimensional chaotic flow systems listed in Sprott [10]. These systems have multiple variables in the ranges of 2 $\sim$ 4, i.e., $N = 2 \sim 4$, for each system. The MATLAB function *ode45* is applied to integrate the differential equations describing the low-dimensional chaotic systems. For the simulation, sampling frequencies have been arbitrarily chosen, ranging from 20 $\sim$ 100 Hz.

In addition to the low-dimensional chaotic systems mentioned above, optimization is further employed on a high-dimensional chaotic flow system that is described by the Mackey–Glass differential equation in Mackey and Glass [26] as follows:

$$\dot{z}(t) = -0.1z(t) + \frac{0.2z(t-\theta)}{1 + z(t-\theta)^{10}} \tag{22}$$

where θ is the time delay and set to 100 herein, as in Cao [13]. The solutions of Eq. (22) are found by the MATLAB function *dde23* and sampled at 100 Hz.

Fig. 12 illustrates the optimization results of the required embedding dimension for the selected systems. In the analysis, signals including Case 1, Airy function [27], $\tanh(t)$ and $\mathrm{sech}(t)$ have been simulated up to the fourth order for

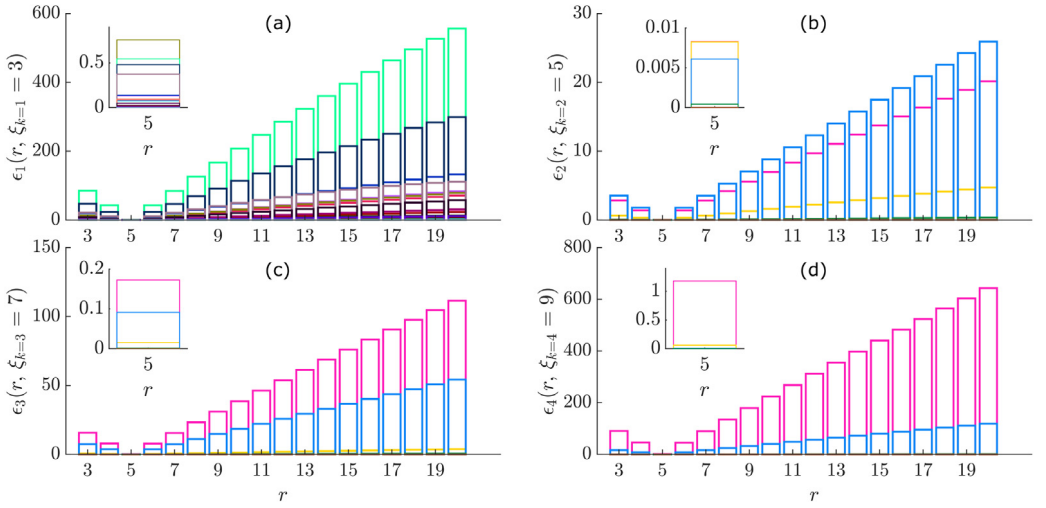

Fig. 12. Required embedding dimension for the best derivative approximation using TDD. Insets show the enlarged view of errors when $r = 5$. Each color represents one system or a signal. (a) represents the error of first-order time derivative estimates for 33 low-dimensional chaotic flow systems and Mackey–Glass delay differential equations. (b) represents the error of second-order time derivatives using the Airy function, the jerk chaotic system I, Case 1, $\tanh(t)$ and $\mathrm{sech}(t)$. (c) represents the error of third-order time derivative estimates using the same signals as in (b). (d) represents the error of fourth-order time derivative estimates using the same signals as in (b) but excluding the jerk chaotic system I due to lack of a simulated fourth-order time derivative.

simplicity. The 33 low-dimensional chaotic flow systems and the Mackey–Glass differential equation have been omitted in the higher-order derivatives because higher-order derivative simulations were not possible for these systems. Because of the same reason, the jerk chaotic system I [28] is only simulated to the third order.

As a result of the optimization, we show that required embedding dimension $\tilde{r} = 5$ always gives the best derivative approximation regardless of the systems/signals chosen. Actually, a smaller embedding dimensions result in underestimated derivatives, whereas higher embedding dimensions lead to overestimated values.

Figs. 13 and 14 show the example optimizations, illustrating the required shifting that minimizes the error in TDD. It is seen that the required shifting data points are 2, 4, 6 and 8 for the first four orders of time derivatives. Theoretically, the SVD-based differentiation and TDD methods yield the averaged derivative for each row of the Hankel matrix. When $r = 5$, the required shifting for SVD-based differentiation should therefore be 2. Analogously, the same shifting is needed for Gibson-II as well. Regarding Gibson-I, the embedding dimension is $r = 3$; therefore, the shifting is 1 for this method. As a result, we show that the shifting for the TDD method is $2k$, where k represents the k^{th} time derivative, which is in agreement with the optimization results, as shown in Fig. 13. In addition, we employ genetic algorithm to check the required embedding dimension by varying the shifting between 0 and 19, and embedding dimension from 3 to 20 to minimize Eq. (18). Results also confirm that the required embedding dimension $\tilde{r}$ is 5 and shifting is $2k$.

5. On the performance of the TDD method in estimating derivatives including higher-order derivatives

5.1. Clean signal

In this section, we systematically test the proposed TDD method for selected cases and compare with some common numerical differentiation techniques, such as the fast Fourier transform (*fft*) and fourth-order central difference (C4D), and with *Gibson-I* and *Gibson-II*. Methods that are described as Gibson refer to Gibson et al. [9], where they use the method of delays combined with Legendre polynomials to estimate the derivatives. *Gibson-I* and *Gibson-II* differ from each other based on the two different constant polynomial coefficients p, where *Gibson-I* refers to $p = 1$ and *Gibson-II* refers to $p = 2$.

Since *Gibson-I* and *Gibson-II* are based on the method of delays, the resulting derivatives should also be shifted. As in the TDD method, we follow the same optimization approach and find the corresponding shifts of 1 and 2 for *Gibson-I* and *Gibson-II*, respectively. Note that Gibson-I is limited to the second time derivative estimation because the Legendre polynomial exists up to the second order for $p = 1$. Similarly, Gibson-II is applicable for time derivative estimations up to the order of 4.

Table 1 illustrates the NRMSE value of the first two derivative approximations for Cases 1 to 4 for clean (without noise) signals. Because the SVD-based differentiation and TDD methods are the same when the first derivative is considered, they are indicated in the same row next to each other in Table 1. As seen, the TDD method generally yields comparable results to the other methods for both the first and second derivative estimations, and the SVD-based second derivative estimation

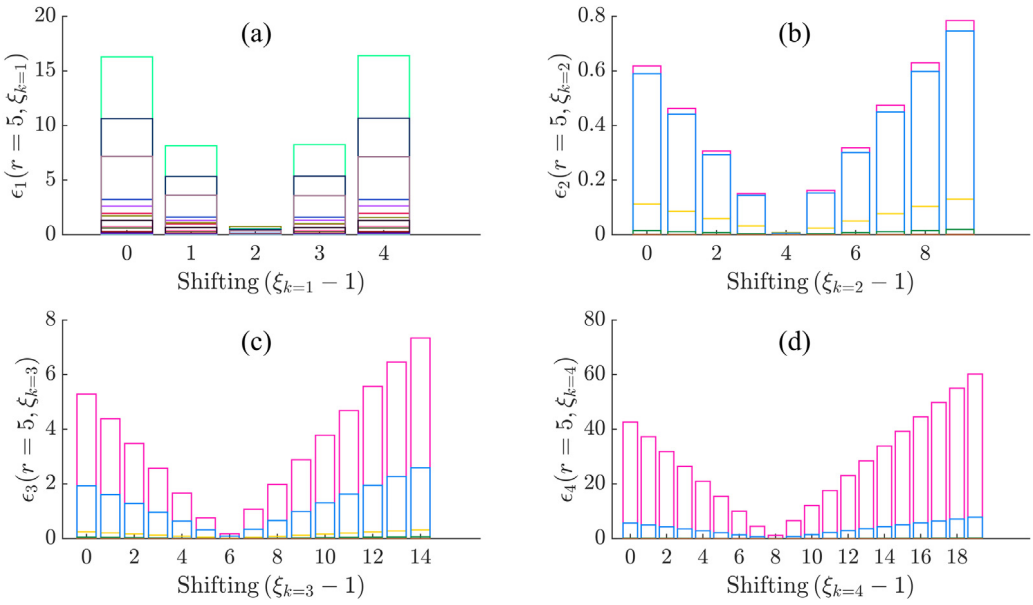

Fig. 13. Required shifting for the best derivative approximation using the TDD method, where the required embedding dimension $\tilde{r} = r = 5$. Each color represents one system and/or a signal. (a) the error of the first-order time derivative estimates for 33 low-dimensional chaotic flow systems and Mackey–Glass delay differential equations. (b) the error of the second time derivative estimate (the TDD method) for Case 1, the Airy function, the jerk chaotic system I and $\tanh(t)$ by shifting $0 \sim 9$ data points. (c) the error of the third time derivative estimate (the TDD method) for signals in (b) by shifting $0 \sim 14$ data points. (d) the error of the fourth time derivative estimate (the TDD method) for signals in (b) excluding the jerk chaotic system I due to lack of a simulated fourth-order time derivative and shift $0 \sim 19$ data points.

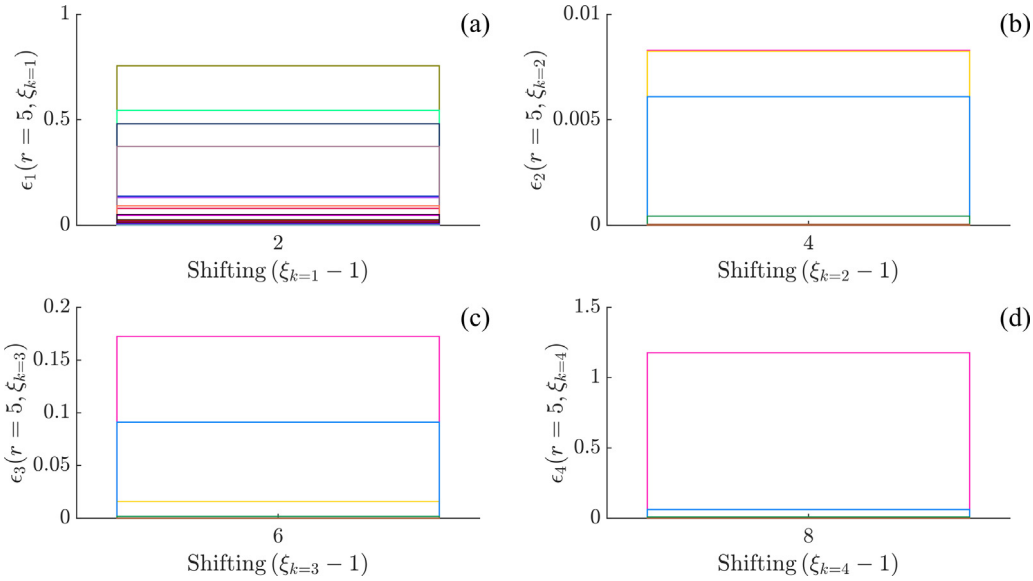

Fig. 14. Required shifting for the best derivative approximation using the SVD-based differentiation method (for the first time derivative only) and the TDD method when using the required embedding dimension ($\tilde{r} = r = 5$). Zoomed-in view of Fig. 13 to show the minimum error. Each color represents one system or a signal. (a) the error of the first-order time derivative estimates for 33 low-dimensional chaotic flow systems and Mackey–Glass delay differential equations. (b) the error of the second time derivative estimate (the TDD method) for Case 1, the Airy function, the jerk chaotic system I and $\tanh(t)$. (c) the error of the third time derivative estimate (the TDD method) for signals in (b). (d) the error of the fourth time derivative estimate (the TDD method) for signals in (b) excluding the jerk chaotic system I due to lack of a simulated fourth-order time derivative and shift $0 \sim 19$ data points.

Table 1
NRMSEs (in %) of estimates of the first two orders of time derivatives for clean signals.

Derivative	Method	Case 1	Case 2	Case 3	Case 4
$\dot{x}$	C4D	1.76×10^{-6}	0.0104	0.0357	0.0321
	fft	0.0331	0.0573	0.0363	0.443
	Gibson-I	4.74×10^{-3}	0.0103	0.0379	0.0348
	Gibson-II	0.0161	0.0106	0.0495	0.0493
	SVD-based/TDD	0.0358	0.0109	0.0494	0.100
$\ddot{x}$	C4D	4.89×10^{-6}	0.106	0.982	0.887
	fft	1.20×10^{-4}	0.109	1.03	0.940
	Gibson-I	3.16×10^{-3}	0.106	1.00	0.912
	Gibson-II	0.0140	0.101	1.02	0.952
	SVD-based	14.9	3.43	3.45	0.963
	TDD	0.0446	0.0949	1.24	1.23

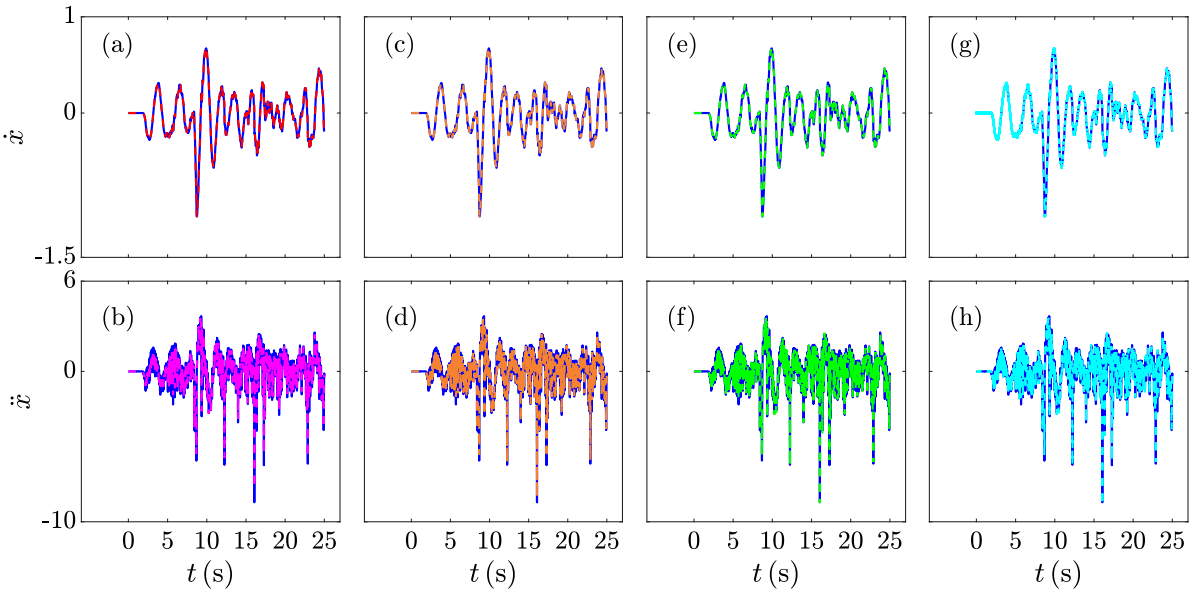

Fig. 15. Comparison between numerical and estimated time derivatives of Case 4. (a–b) represent first- and second-order time derivatives using TDD, (c–d) represent first- and second-order time derivatives using *Gibson-II*, (e–f) represent first- and second-order time derivatives using central fourth-order difference and (g–h) represent first- and second-order time derivatives using *fft*. The blue lines are the simulated results, the red and magenta dashed lines are the results from TDD, the orange dashed line represents the result from *Gibson-II*, the green dashed line represents the result from the central fourth-order derivative, and the cyan dashed line represents the result from *fft*.

is usually not as good as the other methods, as expected. In contrast to all of the other cases, the SVD-based approach produced comparable second-order time derivative estimations only in Case 4 in Table 1.

Fig. 15 shows the comparisons of the first two derivatives obtained by simulation, TDD, *Gibson-II*, C4D and *fft*. As can be seen, no obvious discrepancy exists between the simulated velocity ($\dot{x}$) and the estimates from TDD, C4D and *Gibson-II* combined. However, because the boundary of the signal (not shown here) is not smooth and periodic, a boundary effect on the velocity estimates using *fft* can be clearly seen in Fig. 15g. As a consequence, estimated derivatives have to be truncated at the beginning and end section to eliminate boundary effects as described in Kutz [29]. The results show that all tested methods provide estimates of accelerations that generally agree well with analytical and simulated accelerations.

From an engineering perspective, jerk (derivative of the acceleration) and jounce (derivative of the jerk) are very important parameters (Eager et al. [30], Jazar [31], Smith and Christensen [32]); hence, they should be estimated as accurately as possible. One can also estimate these parameters using TDD following the steps in Fig. 1 for five different systems, including Case 1 (quasi-periodic signal), Airy function as in Oliphant [27], jerk chaotic system I as in Sprott [28], jerk chaotic system II as in Linz and Sprott [33] and the time-varying hyperbolic tangent function.

Table 2 shows the NRMSE value of the third- and fourth-order derivative approximations of the selected cases for clean (without noise) signals. In the analysis, the sampling frequency of the Airy function, the jerk chaotic system I, the jerk chaotic system II and $\tanh(t)$ are chosen as 100 Hz, 20 Hz, 50 Hz and 40 Hz, respectively.

The results of higher-order derivative estimations show one common behavior that TDD successfully computes the third-, fourth- and even higher-order derivatives (not shown). In addition, when some level of noise is added to those signals, as in Table 3, the TDD method performs better at estimating higher-order derivatives.

Table 2
NRMSEs (in %) of estimates of the higher-order time derivatives for clean signals.

Derivative	Method	Case 1	Airy	Jerk I	Jerk II	tanh(t)
$\dddot{x}$	C4D	0.0713	2.88×10^{-5}	0.743	3.16	4.54×10^{-5}
	Gibson-II	0.0721	0.0165	0.663	2.87	0.0115
	TDD	0.0876	0.168	0.342	1.64	0.0615
$\ddddot{x}$	C4D	0.0125	3.62×10^{-5}	N/A	N/A	9.75×10^{-5}
	Gibson-II	0.0151	0.0101	N/A	N/A	9.18×10^{-3}
	TDD	0.0751	0.179	N/A	N/A	0.0965

Table 3
NRMSEs (in %) of estimates of the first two orders of time derivatives for noisy signals (noise level in %)

Derivative	Method	Case 1	Case 2	Case 3	Case 4
		0.1	0.001	0.05	0.01[a]
$\dot{x}$	C4D	0.733	0.123	0.368	0.205
	fft	1.40	0.240	0.701	0.589
	Gibson-I	0.547	0.0913	0.275	0.154
	Gibson-II	0.244	0.0420	0.132	0.0834
	SVD-based/TDD	0.247	0.0420	0.132	0.121
	TVRD[b]	0.0490	0.0301	0.342	0.424
	Tikhonov[c]	0.0615	0.0152	0.0904	0.0666
	DF	0.183	0.0424	0.131	0.0597
$\ddot{x}$	C4D	27.1	8.46	12.8	7.02
	fft	107	33.2	50.1	27.4
	Gibson-I	59.3	18.4	27.8	15.2
	Gibson-II	12.9	4.02	6.16	3.45
	TDD	3.45	1.07	2.01	1.51
	TVRD[b]	0.123	0.208	4.11	4.69
	Tikhonov[c]	0.464	0.214	1.46	1.33
	DF	3.05	0.431	1.48	1.33

[a]This row specifies the noise level in % for each signal.
[b]Abbreviation of total variation regularized differentiation [34].
[c]Shorthand notation for Tikhonov regularization based differentiation [35,36].

5.2. Effect of noise

It is known that real time series are inevitably contaminated by noise; therefore, we further investigate the TDD method in the presence of low-level Gaussian noise for selected cases, where the selected cases include the first four cases (Cases $1 \sim 4$), similar to Table 1.

Subsets of principal components have maximum variance; therefore, they are also expected to have a maximum signal-to-noise ratio, meaning that the power of the signal will be much larger than the embedded noise power. It should also be noted that the TDD method uses a fixed embedding dimension that is found by optimization; therefore, in this sense, PCA becomes the optimal linear coordinate transformation as described in Gibson et al. [9].

Table 3 illustrates the NRMSE value of the first two derivative approximations for Cases 1 to 4 in the presence of noise. As seen, all the methods tested give comparable results when the first-order derivative is considered except *fft*-based differentiation. For the first two derivatives, TVRD outperforms the other methods in Case 1, whereas Tikhonov and DF give the lowest NRMSEs in Case 4. In Cases 2 and 3, Tikhonov is in general superior to other methods. TDD produces similar results with Gibson-II for the first derivative while better results than the latter method for the second derivative. In comparison with DF, TDD yields similar NRMSEs for Case 1 and Case 3. As a result, TDD generally gives comparable estimates of first and second-order derivatives. Another important point here is that, the parameters fed to TVRD (i.e., regularization parameter) and DF (i.e., cut-off frequency and order number of DF) need to be tuned to obtain comparable results in terms of NRMSE. As for Tikhonov, regularization matrix, such as identity matrix, first order derivative operator matrix and second order derivative operator matrix [37–40], is data dependent and should be chosen prior to analysis. Therefore, one need to intuitively know what to expect in the end, which is not always easy. On top of these limitations, the performance of the TVRD and Tikhonov (solved by using LSQR algorithm [41], because of possible large-scale problem encountered) are slow when compared with other methods because of the number of iterations involved to obtain converged results.

Regularization parameter for Tikhonov can be determined by heuristic L-curve method [39,42], generalized cross-validation [43] or other methods mentioned in Hansen [42]. However, none of the existing methods to determine regularization parameter perform well for all signals [42].

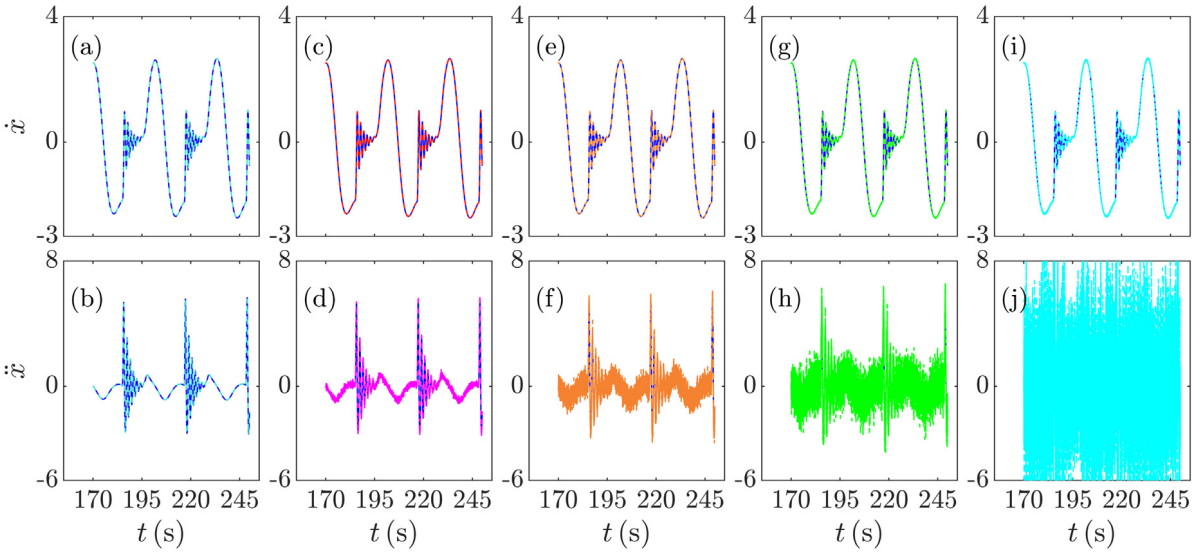

Fig. 16. Comparison between numerical and estimated time derivatives of Case 2. (a–b) represent first and second-order time derivatives using TVRD. (c–d) represent first and second-order time derivatives using TDD, (e–f) represent first- and second-order time derivatives using *Gibson-II*, (g–h) represent first- and second-order time derivatives using C4D and (i–j) represent first- and second-order time derivatives using *fft*. The blue lines are the simulated results, the turquoise dashed lines are results from TVRD, the red and magenta dashed lines are results from TDD, the orange dashed line represents the result from *Gibson-II*, the green dashed line represents the result from C4D, and the cyan dashed line represents the result from *fft*.

The significant effect of the noise on the selected signals can also be seen from the corresponding time history plots. Fig. 16 illustrates the numerical and estimated first- and second-order time derivatives of Case 2 using TVRD, TDD, *Gibson-II*, C4D and *fft*. As seen, TDD produces smoother acceleration signals than the other tested methods except for TVRD that is not cost efficient; therefore, it also produces clearer phase portraits as described in the earlier sections.

6. On the performance of TDD on chaotic systems with multistability

Recently, two studies were performed to explore the dynamic behavior of chaotic systems with multistability [44,45]. One newly devised system described in Natiq et al. [44] is shown below (referred to as Natiq-I hereafter):

$$\dot{y}_1 = -y_2 \tag{23}$$

$$\dot{y}_2 = y_1 + \xi y_2 y_3 \tag{24}$$

$$\dot{y}_3 = \sigma |f_1(y_1)| - \beta y_2^2 - \alpha y_3 - \mu \tag{25}$$

where y_1, y_2, y_3 are state variables; ξ, σ, β, α and μ are system parameters; the control term $f_1(y_1) = \cos(y_1)$.

The second system is a plasma perturbation mathematical model (henceforth denoted as Natiq-II) and is expressed as [45]:

$$\dot{y}_1 = y_2 y_3 - y_2 - \sigma y_1 \tag{26}$$

$$\dot{y}_2 = y_1 \tag{27}$$

$$\dot{y}_3 = \beta\left(\mu - y_3 - y_2^2 y_3\right) + a f_i(y_1),\ i = 1,\ 2. \tag{28}$$

where y_1, y_2, y_3 are state variables; σ, β and μ are independent system parameters; a is a coefficient; the control terms $f_1(y_1) = \exp\left(-y_1^2\right)$ and $f_2(y_1) = \cos(y_1)$. To simulate the Natiq-I and Natiq-II system, ξ, σ, β, μ, α, a, control terms f_i and initial conditions are selected as those presented in Natiq et al. [44,45], see also Appendix. Here, sampling frequency 100 Hz is used for simulation.

Figs. 17–18 exhibit the mean NRMSE (i.e. mean of NRMSE for $\dot{y}_1$, $\dot{y}_2$, and $\dot{y}_3$) of first time derivative estimates by various methods for clean signals, which are produced by simulation, relative to simulated derivatives. Generally, C4D and Gibson-I yield least discrepancy. Gibson-II and TDD give similar results except for Natiq-II cases #21, 22 and 24. However, the deviation of results by using TDD from simulated derivatives is limited within 0.065%. In most of Natiq-I and Natiq-II cases, *fft* produces worst approximation of derivatives, though edge effect is minimized by removing estimations of derivatives in the vicinity of boundary.

By checking the signals (y_1, y_2 and y_3) for Natiq-II cases #21, 22 and 24, we find that signals jump abruptly near initial condition and are strongly damped. In order to improve the performance of TDD for these cases, we propose to

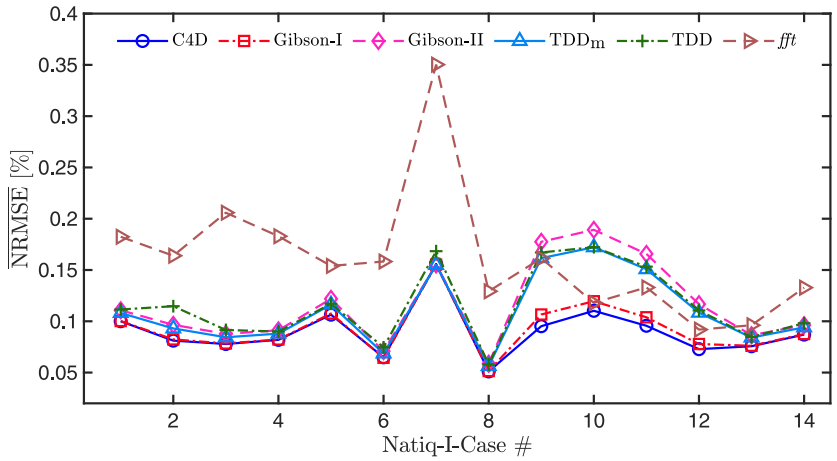

Fig. 17. Mean NRMSE of estimates of first order time derivative for signals produced by simulating Natiq-I system.

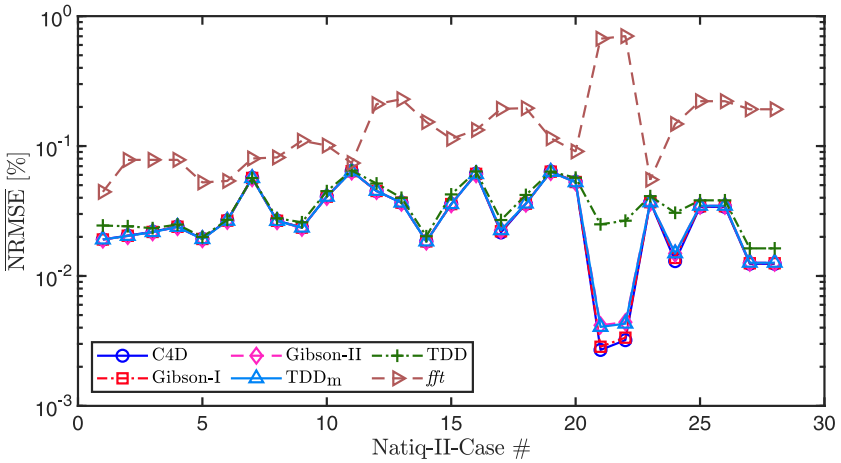

Fig. 18. Mean NRMSE of estimates of first order time derivative for signals produced by simulating Natiq-II system.

mirror the signals, i.e. reverse the original signals and append the reversed signals to the original signals. This way, we not only increase the length of the signal, but also assume that the mirrored signal is periodic. And, after TDD is employed, one can eliminate the mirrored part and obtain the differentiated signal. This signal mirroring technique is similar to zero-phase-shift method for causal filters [46]. The combination of TDD with the signal mirroring technique is referred as mirrored TDD, or simply TDD_m hereafter.

Again, as seen in Figs. 17–18, TDD_m performs either equivalently with or better than TDD and Gibson-II in estimating derivatives.

We also apply TDD_m to Cases $1 \sim 4$. Applying TDD_m on Cases $1 \sim 4$ contaminated by noise (see Table 3), we find that TDD_m yields same or slightly better results when compared with TDD, e.g., NRMSE of $\dot{x}$ decreases from 0.121% to 0.0857% for Case 4. When applied to clean signals (see Table 1), TDD_m results in that NRMSEs of $\dot{\boldsymbol{x}}$ and $\ddot{\boldsymbol{x}}$ drop from 0.0358% to 0.00796% and 0.0446% to 0.0194%, respectively for Case 1. NRMSE of $\dot{\boldsymbol{x}}$ for clean signal of Case 4 after using TDD_m reduces from 0.100% to 0.0520%. For other cases (Case 2 and Case 3), TDD_m gives the identical results as TDD.

TDD_m reduces NRMSE of $\dot{\boldsymbol{x}}$ of Case 4 (both clean and noisy signals) due to that $\boldsymbol{x}$ has quasi-linear drift component (see Fig. 8a) and TDD_m manages to handle traveling motions better than TDD (see the reason in Section 7).

In terms of third and fourth order time derivatives (see Table 2), TDD_m improves the estimations for Airy function markedly. More specifically, NRMSE of $\dddot{\boldsymbol{x}}$ decreases from 0.168% to 0.0475% and NRMSE of $\ddddot{\boldsymbol{x}}$ declines from 0.179% to 0.0560%. Similar to Natiq-II cases #21, 22 and 24, time histories of second and third derivatives of Airy function are decaying in time. This is why TDD_m gives rise to better estimates of higher order derivatives than TDD for Airy function.

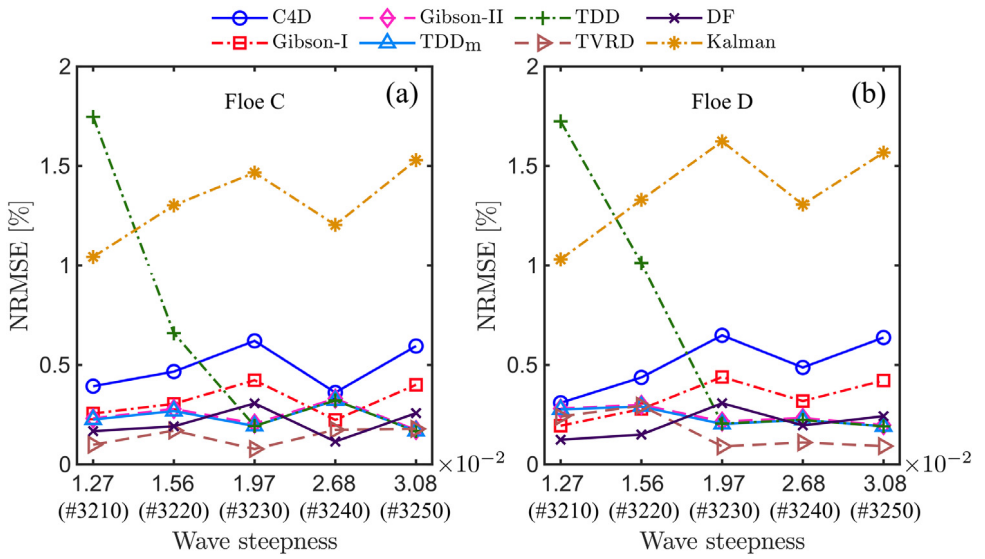

Fig. 19. NRMSEs (with respect to results obtained by using Tikhonov) of estimating velocities of ice floes C and D in test runs #3210 ∼ #3250.

7. On the performance of TDD and its variant on real experimental signals

Here, we apply TDD and TDD_m together with other methods on experimental signals obtained from the same experimental campaign described in Section 3.2.

Velocities of floes C and D in test runs #3210 ∼ #3250 are quantified by using Kalman filtering (Smyth and Wu [17],Faragher [47], hereafter Kalman), C4D and DF in a recent study [25].

As Tikhonov in general produces high-quality estimates of derivatives for noisy signals (see Table 3), we compare results from other differentiation methods with those given by using Tikhonov.

Deviations of the results obtained by using various differentiation methods from those by Tikhonov are summarized in Fig. 19. For all experimental signals, Kalman gives results that deviate most significantly from those by means of Tikhonov (NRMSE larger than 1%). The main reason is that the motion of floes C and D differs slightly from ice floes that were instrumented with IMUs (see details in Li et al. [25]). However, Kalman optimally combines displacement measurements with acceleration measurements. Hence, the results given by Kalman are quality-assurance for estimates produced by other methods [25].

TDD yields evidently different estimates of velocities for test runs #3210 and #3220 in comparison with Tikhonov. This is attributed to quasi-linear drift motion of ice floes C and D in these two test runs (see Figs. 10c and 11c). TDD is based on SVD, which has inherent caveat for traveling motion signals [48]. In contrast, TDD_m improves the estimates appreciably for test runs #3210 and #3220.

For all motion signals of ice floes studied here, TDD_m gives similar estimates of velocities with the methods C4D, Gibson-I, Gibson-II, Tikhonov, TVRD and DF (NRMSE within 0.65%).

Another set of experimental signals analyzed here are ECG signals [49–51]. These ECG signals include 17 states measured on 45 patients, and are sampled at 360 Hz with a duration of 10 s [49]. We randomly choose 17 signals and each of them corresponds with a different state. The results are displayed in Fig. 20. It is seen that results obtained by means of TDD and TDD_m are equivalent and deviate least from those produced by Tikhonov in general (NRMSE less than 1.5%). In contrast, C4D results in estimates that have largest discrepancy from the results given by Tikhonov. The other methods (Gibson-I, Gibson-II, TVRD and DF) yield derivatives with modest difference from Tikhonov.

8. Summary and conclusion

In this study, we introduced a simple analytical function based on the method of delays with PCA to estimate time derivatives of a given signal. In the function, we showed that the sampling frequency term scales the derivatives in PCA correctly and estimates the higher-order derivatives accurately when compared with some common numerical differentiators.

It is known that time derivative estimations change with the embedding dimension in the delay coordinates; however, when TDD is considered, we showed that the required embedding dimension that best describes the analytical derivatives is a constant value, which is equal to 5. This is particularly significant, given the simplicity of the method. In addition, unlike Gibson et al. [9], TDD does not require creating complex polynomials to estimate higher-order derivatives because we update the Hankel matrix in each derivation.

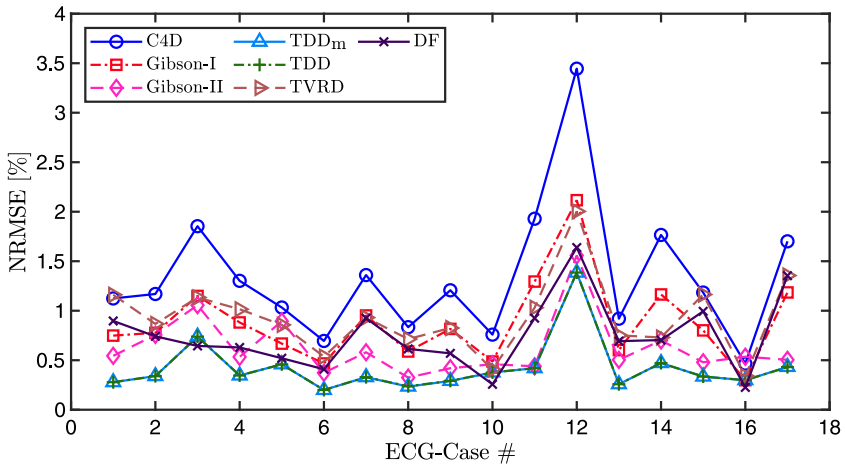

Fig. 20. NRMSE (with respect to results obtained by using Tikhonov) of estimates of first order time derivative for ECG signals.

Using numerical (including 38 different chaotic systems) and experimental cases, we have demonstrated the applicability of TDD to different types of signals in both the absence and the presence of some level of Gaussian noises. At the end, we have shown that TDD and its variant TDD_m produce comparable results in the absence of noise and computes the derivatives better in the presence of low-level Gaussian noise.

We conclude that there is not a single method that works best in all applications in terms of accurate and fast numerical differentiation. Depending on the application, noise level and many other parameters, one method might outperform the other. Therefore, this process is really application dependent. Some methods such as total variation regularized differentiation (TVRD), Tikhonov regularization based differentiation (Tikhonov), differentiator filter (DF) either need to be tuned or a regularization matrix should be chosen to obtain comparable or better results; which means, one need to intuitively know what to expect as a result priori and this is not always easy. Also, the performances of the TVRD and Tikhonov are slow when compared with other methods that are mentioned in this paper. However, one consistent result is that the new TDD (and TDD_m) method always gives comparable results to estimate the derivatives of a signal when compared with other well established methods such as central-fourth order difference, TVRD, and Gibson's Legendre polynomials and therefore it can be used as a new numerical differentiaton method.

CRediT authorship contribution statement

Hongtao Li: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Writing - original draft, Writing - review & editing, Visualization. **Ersegun Deniz Gedikli:** Conceptualization, Methodology, Writing - original draft, Writing - review & editing, Resources. **Raed Lubbad:** Writing - review & editing, Funding acquisition, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The authors wish to acknowledge the support from the Research Council of Norway through the Centre for Research-Based Innovation SAMCoT, and the support from all SAMCoT partners. The experimental work described in this publication was supported by the European Community's Horizon 2020 Programme through the grant to HYDRALAB+, contract 654110. The lead author would like to thank the Hamburg Ship Model Basin (HSVA), especially the ice tank crew, for their hospitality, the technical and scientific support, and the professional execution of the test program in the Research Infrastructure ARCTECLAB. The lead author would like to extend thanks to Hayley Shen and Meleta Truax from Clarkson University, Agnieszka Herman from the University of Gdansk, Sergiy Sukhorukov from Kvaerner AS, Karl-Ulrich Evers from HSVA and Sveinung Lset and Andrei Tsarau from the Norwegian University of Science and Technology for their help during the test. The first author would like to thank Zhengshun Cheng from the Department of Marine Technology at NTNU (now at Shanghai Jiao Tong University) for inspiring discussions. The lead author also thanks Per Christian Hansen, and Yiqiu Dong at DTU Compute for constructive discussions, which happened when the lead author stayed at DTU.

Table A.1
Parameter values and initial conditions for Natiq-I system [44].

Natiq-I-Case #	ξ	σ	β	α	μ	Initial condition
1	0.63	17	0.47	0.5	0	(2, −2, −1)
2	0.63	17	0.47	0.5	0	(0.5, −2, 0)
3	0.63	17	1.4	0.5	0	(1.8, 0.3, 0)
4	0.63	17	1.4	0.5	0	(−1.8, 0.3, 0)
5	0.63	17	1.4	0.5	0	(0.5, −2, 0)
6	0.63	17	1.93	0.5	0	(2, −2, −1)
7	0.63	17	1.93	0.5	0	(1.8, 0.3, 0)
8	0.63	17	1.93	0.5	0	(0.5, −2, 0)
9	1.82	12	0.1	0	0	(1.8, 0.3, 0)
10	1.82	12	0.1	0	0	(−1.8, 0.3, 0)
11	1.82	12	0.1	0	0	(5, 0, −5)
12	1.82	12	0.1	0	0	(5, 3, −3)
13	0.63	17	0.3	0.63	0.3	(1, −1, 1)
14	0.63	17	0.3	0.63	0.3	(1, −1, −4)

Table A.2
Parameter values and initial conditions for Natiq-II system [45].

Natiq-II-Case #	σ	β	μ	a	f_i	Initial condition
1	0.5	0.1	2.2	0	0	(1, 0.5, 1)
2	0.5	0.1	2.2	0.5	$\exp\left(-y_1^2\right)$	(−2, 0.01, −1)
3	0.5	0.1	2.2	0.76	$\exp\left(-y_1^2\right)$	(−2, 0.01, −1)
4	0.5	0.1	2.2	0.8	$\exp\left(-y_1^2\right)$	(−2, 0.01, −1)
5	0.5	0.1	2.2	0.5	$\cos(y_1)$	(−2, 0.099, −1)
6	0.5	0.1	2.2	0.76	$\cos(y_1)$	(−2, 0.099, −1)
7	0.5	0.1	2.2	0.85	$\cos(y_1)$	(−2, 0.099, −1)
8	0.5	0.1	2.2	0.8	$\exp\left(-y_1^2\right)$	(−2, 0.7, −1)
9	0.5	0.1	2.2	0.8	$\exp\left(-y_1^2\right)$	(−2, 2, −1)
10	0.5	0.1	2.2	0.85	$\cos(y_1)$	(−2, 0.5, −1)
11	0.5	0.1	2.2	0.85	$\cos(y_1)$	(−2, 1.2, −1)
12	0.5	0.1	2.2	0.85	$\cos(y_1)$	(−2, 2, −1)
13	0.5	0.1	2.2	0.85	$\cos(y_1)$	(−2, 3, −1)
14	0.5	0.09	2.5	0.7	$\exp\left(-y_1^2\right)$	(2, 1.6, −2)
15	0.5	0.09	2.5	0.7	$\exp\left(-y_1^2\right)$	(2, 1.6, 2)
16	0.5	0.09	2.5	0.72	$\cos(y_1)$	(2, 4, 2)
17	0.5	0.09	2.5	0.72	$\cos(y_1)$	(−2, 4, −2)
18	0.5	0.09	2.5	0.72	$\cos(y_1)$	(−2, 1.5, −2)
19	0.5	0.09	2.5	0.72	$\cos(y_1)$	(−2, −1.5, −2)
20	0.45	0.6	2	2	$\cos(y_1)$	(1, 1, 3)
21	0.45	0.6	2	2	$\cos(y_1)$	(1, 4.4, 3)
22	0.45	0.6	2	2	$\cos(y_1)$	(1, −3.9, 3)
23	0.45	0.6	2	1.02	$\cos(y_1)$	(1, 1, 3)
24	0.45	0.6	2	1.02	$\cos(y_1)$	(1, −3.9, 3)
25	0.5	0.69	1	0.52	$\cos(y_1)$	(0, 1, 0)
26	0.5	0.69	1	0.52	$\cos(y_1)$	(0, −1, 0)
27	0.5	0.71	1	0.52	$\cos(y_1)$	(0, 1, 0)
28	0.5	0.71	1	0.52	$\cos(y_1)$	(0, −1, 0)

Appendix. Parameter values and initial conditions for chaotic systems with multistability

Parameter values and initial conditions for Natiq-I and Natiq-II system are listed in Table A.1 and Table A.2, respectively.

References

[1] N.H. Packard, J.P. Crutchfield, J.D. Farmer, R.S. Shaw, Geometry from a time series, Phys. Rev. Lett. 45 (9) (1980) 712.
[2] F. Takens, Detecting strange attractors in turbulence, in: Dynamical Systems and Turbulence, Warwick 1980, Springer, 1981, pp. 366–381.
[3] H. Yu, X. Li, On the chaos analysis and prediction of aircraft accidents based on multi-timescales, Physica A 534 (2019) 120828.

[4] D.S. Broomhead, G.P. King, Extracting qualitative dynamics from experimental data, Physica D 20 (2–3) (1986) 217–236.
[5] P. Shang, A. Lin, L. Liu, Chaotic SVD method for minimizing the effect of exponential trends in detrended fluctuation analysis, Physica A 388 (5) (2009) 720–726.
[6] G. Kerschen, J.-C. Golinval, A. Vakakis, L. Bergman, The method of proper orthogonal decomposition for dynamical characterization and order reduction of mechanical systems: an overview, Nonlinear Dynam. 41 (2005) 147–169.
[7] E.D. Gedikli, D. Chelidze, J. Dahl, Observed mode shape effects on the vortex-induced vibration of bending dominated flexible cylinders simply supported at both ends, J. Fluid Struct. 81 (2018) 399–417.
[8] E.D. Gedikli, T. Nord, H. Hendrikse, G. Ziemer, On pressure modes in ice-induced vibrations using multivariate analysis, Cold Reg. Sci. Technol. 160 (2019) 150–162.
[9] J.F. Gibson, J. Doyne Farmer, M. Casdagli, S. Eubank, An analytic approach to practical state space reconstruction, Physica D 57 (1–2) (1992) 1–30.
[10] J.C. Sprott, Chaos and Time-Series Analysis, Vol. 69, Oxford University Press, 2003.
[11] A. Chatterjee, An introduction to the proper orthogonal decomposition, Curr. Sci. (2000) 808–817.
[12] M.B. Kennel, R. Brown, H.D. Abarbanel, Determining embedding dimension for phase-space reconstruction using a geometrical construction, Phys. Rev. A 45 (6) (1992) 3403.
[13] L. Cao, Practical method for determining the minimum embedding dimension of a scalar time series, Physica D 110 (1–2) (1997) 43–50.
[14] M.B. Kennel, H.D. Abarbanel, False neighbors and false strands: A reliable minimum embedding dimension algorithm, Phys. Rev. E 66 (2) (2002) 026209.
[15] A. Maus, J. Sprott, Neural network method for determining embedding dimension of a time series, Commun. Nonlinear Sci. Numer. Simul. 16 (8) (2011) 3294–3302.
[16] D. Chelidze, Reliable estimation of minimum embedding dimension through statistical analysis of nearest neighbors, J. Comput. Nonlinear Dyn. 12 (5) (2017) 051024.
[17] A. Smyth, M. Wu, Multi-rate Kalman filtering for the data fusion of displacement and acceleration response measurements in dynamic system monitoring, Mech. Syst. Signal Process. 21 (2) (2007) 706–723.
[18] STRONG-MOTION VIRTUAL DATA CENTER, Global component of the center for engineering strong motion data, 2018, https://strongmotioncenter.org/vdc/scripts/default.plx (Accessed: 24 May 2018).
[19] A. Tsarau, Data storage plan-project HY+_HSVA-01 Expreimental study on wave propagation in ice and the combined action of waves and ice on structures "Loads on Structures - Waves propagating in ice" LS-WICE, 2017, URL https://doi.org/10.5281/zenodo.1067170, LIMB - Hamburg Ship Model Basin.
[20] S. Cheng, A. Tsarau, H. Herman, K.U. Evers, H.H. Shen, Loads on Structure and Waves in Ice (LS-WICE) project. Part 1: Wave attenuation and dispersion in broken ice fields, in: Proceedings of the 24th International Conference on Port and Ocean Engineering under Arctic Conditions, POAC, Busan, Korea, 2017.
[21] H. Herman, A. Tsarau, K.U. Evers, H. Li, H.H. Shen, Loads on Structure and Waves in Ice (LS-WICE) project. Part 2: Sea ice breaking by waves, in: Proceedings of the 24th International Conference on Port and Ocean Engineering under Arctic Conditions, POAC, Busan, Korea, 2017.
[22] A. Tsarau, S. Sukhorukov, H. Herman, K.U. Evers, S. Lset, Loads on Structure and Waves in Ice (LS-WICE) project. Part 3: Ice-structure interaction under wave actions, in: Proceedings of the 24th International Conference on Port and Ocean Engineering under Arctic Conditions, POAC, Busan, Korea, 2017.
[23] H. Li, R. Lubbad, Laboratory study of ice floes collisions under wave action, in: Proceedings of the 28th International Ocean Polar Engineering Conference, ISOPE, Sapporo, Japan, 2018, pp. 1516–1524.
[24] MATLAB, Take derivatives of a signal, 2017, https://se.mathworks.com/help/signal/ug/take-derivatives-of-a-signal.html (Accessed: 27 March 2017).
[25] H. Li, E.D. Gedikli, R. Lubbad, T.S. Nord, Laboratory study of wave-induced ice-ice collisions using robust principal component analysis and sensor fusion, Cold Reg. Sci. Technol. 172 (2020) 103010.
[26] M.C. Mackey, L. Glass, Oscillation and chaos in physiological control systems, Science 197 (4300) (1977) 287–289.
[27] T.E. Oliphant, A Guide to NumPy, Vol. 1, Trelgol Publishing USA, 2006.
[28] J.C. Sprott, Simplest dissipative chaotic flow, Phys. Lett. A 228 (4–5) (1997) 271–274.
[29] J. Kutz, Data-Driven Modeling & Scientific Computation: Methods for Complex Systems & Big Data, Oxford University Press, 2013.
[30] D. Eager, A.-M. Pendrill, N. Reistad, Beyond velocity and acceleration: jerk, snap and higher derivatives, Eur. J. Phys. 37 (6) (2016) 065008.
[31] R.N. Jazar, Advanced Dynamics: Rigid Body, Multibody, and Aerospace Applications, John Wiley & Sons, 2011.
[32] C. Smith, H.I. Christensen, A minimum jerk predictor for teleoperation with variable time delay, in: Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on, IEEE, 2009, pp. 5621–5627.
[33] S.J. Linz, J. Sprott, Elementary chaotic flow, Phys. Lett. A 259 (3–4) (1999) 240–245.
[34] R. Chartrand, Numerical differentiation of noisy, nonsmooth data, ISRN Appl. Math. 2011 (2011).
[35] J. Cullum, Numerical differentiation and regularization, SIAM J. Numer. Anal. 8 (2) (1971) 254–265.
[36] I. Knowles, R.J. Renka, Methods for numerical differentiation of noisy data, Electron. J. Differ. Equ. Conf. 21 (2014) 235–246.
[37] Z. Chen, T.H. Chan, A truncated generalized singular value decomposition algorithm for moving force identification with ill-posed problems, J. Sound Vib. 401 (2017) 297–310.
[38] E.D. Gedikli, D. Chelidze, J.M. Dahl, Bending dominated flexible cylinder experiments reveal insights into modal interactions for flexible body vortex-induced vibrations, in: Proceedings of the 28th International Ocean Polar Engineering Conference, ISOPE, Sapporo, Japan, 2018.
[39] J.L. Mueller, S. Siltanen, Linear and Nonlinear Inverse Problems with Practical Applications, Vol. 10, SIAM, 2012.
[40] E.D. Gedikli, D. Chelidze, J. Dahl, Empirical mode analysis identifying hysteresis in vortex-induced vibrations of a bending-dominated flexible cylinder, Int. J. Offshore Polar. 30 (02) (2020) 186–193.
[41] C.C. Paige, M.A. Saunders, LSQR: An algorithm for sparse linear equations and sparse least squares, ACM Trans. Math. Softw. (TOMS) 8 (1) (1982) 43–71.
[42] P.C. Hansen, Discrete Inverse Problems: Insight and Algorithms, Vol. 7, SIAM, 2010.
[43] R.C. Aster, B. Borchers, C.H. Thurber, Parameter Estimation and Inverse Problems, Elsevier, 2018.
[44] H. Natiq, M. Said, M. Ariffin, S. He, L. Rondoni, S. Banerjee, Self-excited and hidden attractors in a novel chaotic system with complicated multistability, Eur. Phys. J. Plus 133 (12) (2018) 1–12.
[45] H. Natiq, S. Banerjee, A. Misra, M. Said, Degenerating the butterfly attractor in a plasma perturbation model using nonlinear controllers, Chaos Solitons Fractals 122 (2019) 58–68.
[46] M.X. Cohen, Analyzing Neural Time Series Data: Theory and Practice, MIT press, 2014.
[47] R. Faragher, Understanding the basis of the Kalman filter via a simple and intuitive derivation, IEEE Signal Process. Mag. 29 (5) (2012) 128–132.
[48] S.L. Brunton, J. Kutz, Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control, Cambridge University Press, 2019.
[49] P. Plawiak, ECG signals (1000 fragments), 2017, URL http://dx.doi.org/10.17632/7dybx7wyfn.3 (Accessed : 09 February 2020).
[50] G.B. Moody, R.G. Mark, The impact of the MIT-BIH arrhythmia database, IEEE Eng. Med. Biol. Mag. 20 (3) (2001) 45–50.
[51] A.L. Goldberger, L.A. Amaral, L. Glass, J.M. Hausdorff, P.C. Ivanov, R.G. Mark, J.E. Mietus, G.B. Moody, C.-K. Peng, H.E. Stanley, PhysioBank, PhysioToolkit, and PhysioNet: components of a new research resource for complex physiologic signals, Circulation 101 (23) (2000) e215–e220.