applied sciences

MDPI

Article

A Novel Neural Network with the Ability to Express the Extreme Points Distribution Features of Higher Derivatives of Physical Processes

Xibo Wang *, Feiyan Ma, Yanfei Gao *, Jinfeng Liang and Changfeng Zhou

Department of Automotive Engineering, Key Laboratory of Transportation Vehicle Testing, Diagnosis and Maintenance Technology (Jinan), Shandong Jiaotong University, Jinan 250357, China; mfeiyan2022@163.com (F.M.)

* Correspondence: wangxibo@sdjtu.edu.cn (X.W.); gaoyanfei@sdjtu.edu.cn (Y.G.)

Abstract: Higher derivatives are important to interpret the physical process. However, higher derivatives calculated from measured data often deviate from the real ones because of measurement errors. A novel method for data fitting without higher derivatives violating the real physical process is developed in this paper. Firstly, the research on errors' influence on higher derivatives and the typical functions' extreme points distribution were conducted, which demonstrates the necessity and feasibility of adopting extreme points distribution features in neural networks. Then, we proposed a new neural network considering the extreme points distribution features, namely, the extreme-points-distribution-based neural network (*EDNN*), which contains a sample error calculator (*SEC*) and extreme points distribution error calculator (*EDEC*). With recursive automatic differentiation, a model calculating the higher derivatives of the *EDNN* was established. Additionally, a loss function, embedded with the extreme points distribution features, was introduced. Finally, the *EDNN* was applied to two specific cases to reduce the noise in a second-order damped free oscillation signal and an internal combustion engine cylinder pressure trace signal. It was found that the *EDNN* could obtain higher derivatives that are more compatible with physical trends without detailed differentiation equations. The standard deviation of derivatives' error of the *EDNN* is less than 62.5 percent of that of traditional neural networks. The *EDNN* provides a novel method for the analysis of physical processes with higher derivatives compatible with real physical trends.

Keywords: higher derivatives; extreme-points-distribution-based neural network; recursive automatic differentiation; noise reduction; cylinder pressure trace; second-order damped free oscillation

Citation: Wang, X.; Ma, F.; Gao, Y.; Liang, J.; Zhou, C. A Novel Neural Network with the Ability to Express the Extreme Points Distribution Features of Higher Derivatives of Physical Processes. *Appl. Sci.* **2023**, *13*, 6662. https://doi.org/10.3390/app13116662

Academic Editors: Alexander N. Pisarchik and Yoshiyasu Takefuji

Received: 1 April 2023
Revised: 4 May 2023
Accepted: 26 May 2023
Published: 30 May 2023

1. Introduction

Higher derivatives are important to theoretical research and engineering application [1–6] for deepening the understanding of physical processes and improving analysis efficiency [7,8]. However, it is difficult to obtain higher derivatives directly from a real signal because of measurement errors [9]. Though varied techniques such as the Savitzky–Golay polynomial [10,11], Fourier transform [12,13] and wavelet transform [14,15] have been developed to obtain higher derivatives of measured signals, the higher derivatives obtained with the above techniques often deviate from the real one in trends. It is a challenge and urgent demand [9] to develop a data fitting method without higher derivatives violating real trends.

With the development of physics-informed neural networks (*PINN*) [16–18] and the advancement in automatic differentiation [19–22], complex nonlinear feature recognition [23–26] and parameter identification of physical processes with well-defined differential equations [27,28] were extensively studied, which demonstrate the neural networks' ability to obtain higher derivatives.

Though neural networks can fit any continuous function [29–31], the higher-derivatives-constrained *PINN* could reduce the disturbance of noise [32–34] in data fitting. However, when lacking detailed mathematical models, the relations between the derivatives could not be established, and *PINN* could not be applied in fitting those physical processes.

In order to fit the measured signal without a discrepancy in the trend of higher derivatives, stemming from the local monotonicity of the real physical process [35,36], the necessity and feasibility of adopting the number of extreme points as constraints to data fitting with neural networks are proved. Then, an extreme-points-distribution-based neural network (*EDNN*) is proposed and applied in noise reduction. The research indicates that the *EDNN* could obtain higher derivatives that are more compatible to physical trends without detailed differential relations and could be applied in the denoising of measured signal with higher derivatives conforming to real physical trends. The proposed *EDNN* provides a novel way for signal denoising that retains higher derivatives compatible with real physical process trends.

2. Analysis of Noise's Influence on Higher Derivatives and Extreme Points Distribution of Typical Process Functions

2.1. Noise's Influence on the Higher Derivatives of the Measured Signal

Generally, the signal to be measured $f(t)$ is continuous in the time domain. In the neighborhood of t_0, it can be approximated using a Taylor series as in expression (1).

$$\begin{aligned} f(t) = f(t_0) + f'(t_0)(t-t_0) + \tfrac{1}{2!}f''(t_0)(t-t_0)^2 + \cdots + \tfrac{1}{n!}f^{(n)}(t_0)(t-t_0)^n \\ +o((t-t_0)^n) \end{aligned} \tag{1}$$

Suppose that there is a continuous noise $\varepsilon(t)$ in the vicinity of t_0; the noise $\varepsilon(t)$ can also be approximated using a Taylor series as in expression (2).

$$\begin{aligned} \varepsilon(t) = \varepsilon(t_0) + \varepsilon'(t_0)(t-t_0) + \tfrac{1}{2!}\varepsilon''(t_0)(t-t_0)^2 + \cdots + \tfrac{1}{n!}\varepsilon^{(n)}(t_0)(t-t_0)^n \\ +o((t-t_0)^n) \end{aligned} \tag{2}$$

The signal to be measured and noise can be superposed together. Then, the measured signal $C(t)$ in the vicinity of t_0 can be approximated using a Taylor series as in expression (3).

$$\begin{aligned} &C(t) = f(t) + \varepsilon(t) \\ &= f(t_0) + f'(t_0)(t-t_0) + \tfrac{1}{2!}f''(t_0)(t-t_0)^2 + \cdots \\ &+\tfrac{1}{n!}f^{(n)}(t_0)(t-t_0)^n + o((t-t_0)^n) + \varepsilon(t_0) + \varepsilon'(t_0)(t-t_0) \\ &+\tfrac{1}{2!}\varepsilon''(t_0)(t-t_0)^2 + \cdots + \tfrac{1}{n!}\varepsilon^{(n)}(t_0)(t-t_0)^n + o((t-t_0)^n) \end{aligned} \tag{3}$$

Because the noise $\varepsilon(t)$ is random, there exists the probability that the first k terms of the Taylor series at t_0 are zero and the term after the $k+1$ term can be ignored; thus:

$$\begin{aligned} \varepsilon(t) \quad &= \varepsilon(t_0) + \varepsilon'(t_0)(t-t_0) + \tfrac{1}{2!}\varepsilon''(t_0)(t-t_0)^2 + \cdots + \tfrac{\varepsilon(t_t)}{(t_t-t_0)^k}(t-t_0)^k \\ &+ \cdots \tfrac{1}{n!}\varepsilon^{(n)}(t_0)(t-t_0)^n + o((t-t_0)^n) = \tfrac{1}{k!}\varepsilon^{(k)}(t_0)(t-t_0)^k \end{aligned} \tag{4}$$

Suppose the noise amplitude at the time t_t is ε_t, i.e., $\varepsilon(t_t) = \varepsilon_t$, then:

$$\varepsilon(t_t) = \frac{1}{k!}\varepsilon^{(k)}(t_0)(t_t-t_0)^k \tag{5}$$

$$\varepsilon^{(k)}(t_0) = \frac{\varepsilon(t_t)}{\frac{1}{k!}(t_t-t_0)^k} = \frac{\varepsilon_t}{\frac{1}{k!}(t_t-t_0)^k} \tag{6}$$

For $\forall A_\varepsilon$, when $|\varepsilon_t| > \left|A_\varepsilon \cdot \frac{1}{k!}(t_t-t_0)^k\right|$, $\exists\left|\varepsilon^{(k)}(t_0)\right| > |A_\varepsilon|$.

Therefore, when the noise amplitude ε_t is a constant, as t_t approaches t_0, $\frac{\varepsilon(t_t)}{(t_t-t_0)^k}$ will become larger and larger, as will $\varepsilon^{(k)}(t_0)$. That is, the kth order derivative of the noise at t_0

becomes larger and larger. Subsequently, the kth order derivative of the measured signal $C(t)$ also becomes larger and larger. When ε_t is large enough or $(t_t - t_0)$ is small enough, the kth order derivative of noise $\varepsilon(t)$ at t_0 will be large enough, and the kth order derivative of measured signal $C(t)$ at t_0 will deviate seriously from the corresponding derivative of the signal $f(t)$.

The above analysis shows that even if the Taylor series expansion of the noise has only the kth order term, provided the kth order derivative is large enough, the kth order derivative of the measured signal can be seriously interfered with. Additionally, the kth order derivative of the measured signal will deviate substantially from the kth order derivative of the real physical process. Therefore, it is necessary to reduce the noise disturbance to obtain the higher derivatives of the measured data conforming to the real physical trend.

2.2. Extreme Points Distribution of Typical Process Functions

When fitting a measured signal using neural networks, noise often leads to the higher derivatives of the fitted signal to deviate from the true physical trend [37].

Generally, the relation between the physical quantities and time of a physical process can be modeled by a polynomial, Gaussian function, cosine function, or exponential function. The extreme points of those functions' higher derivatives on a specific interval are distributed in certain patterns. When extreme points distribution is involved in data fitting, the approximation between the higher derivatives of the measured signal and the higher derivatives of the real physical process could be improved. It is necessary to analyze the extreme points' distribution pattern of those typical process functions and their higher derivatives to involve the extreme points distribution in data fitting with neural networks.

An nth order polynomial modeling a continuous process is shown in expression (7).

$$p(x) = \sum_{i=0}^{n} a_i \cdot x^i \tag{7}$$

Additionally, the kth order derivative of an nth order polynomial can be obtained in (8).

$$p^{(k)}(x) = k!a_k + \frac{(k+1)!}{1!}a_{(k+1)}x + \frac{(k+2)!}{2!}a_{(k+2)}x^2 + \cdots + \frac{n!}{(n-k)!}a_n x^{n-k} \tag{8}$$

It can be seen that the kth order derivative of nth order polynomial is an $(n-k)$th-order polynomial. Therefore, the number of the kth-order derivative's extreme points decreases with the increase of order k. When fitting a measured signal modeled by an nth-order polynomial, the extreme point number could be applied to test the consistency between the fitted kth order derivative and that of the real physical process.

An exponential function describing the growth or decline process is shown in (9).

$$f(x) = a^x \tag{9}$$

Additionally, the kth-order derivative of the exponential function is shown in (10).

$$f^{(k)}(x) = [ln(a)]^k a^x \tag{10}$$

According to (10), the number of extreme points of the 0–kth-order derivatives of the exponential function is always zero. When fitting a measured signal modeled using the exponential function, the number of extreme points should always remain zero. Any extreme point appearing in the fitting results indicates a deviation from the physical trend.

A Gaussian function describing a random process is shown in expression (11).

$$g(x) = e^{ax^2} \tag{11}$$

The kth-order derivative of the Gaussian function is shown in expression (12).

$$\frac{d^n g(x)}{dx^n} = e^{ax^2} \sum_{i=0}^{n} C_{it}^{n}(a) x^m \tag{12}$$

where $C_{it}^{n}(a) = \frac{(2a)^t}{1+i\varepsilon_n^+} F_i(n)[(n-i) - \varepsilon_n^+]!!$, $m = i + \varepsilon_n^-$, $\varepsilon_n^{\pm} = \frac{1}{2}\left[1 \pm (-1)^n\right]$, $t = \frac{(n+i)}{2} + \varepsilon_n^-$ and $F_i(n) = \frac{n!}{i!(n-i)!}$ is the binomial coefficient.

The highest degree of x^m in expression (12) increases as the derivative order increases [38]. Then, the number of the derivative's extreme points increases as the derivative order increases. When fitting a measured signal modeled using the Gaussian function, the extreme point number could be applied to test the consistency between the fitted kth-order derivative and that of the real physical process.

The cosine function describing a simple periodic process is shown in (13).

$$h(x) = cos(x) \tag{13}$$

The kth-order derivative of the cosine function is shown in expression (14).

$$h^{(k)}(x) = cos(\frac{k\pi}{2} + x) \tag{14}$$

Cosine function shifted left by $k\pi/2$ yields its kth-order derivative. If the number of extreme points of the cosine function is N_{hp} on any arbitrary interval $[-a, +a]$, the number of extreme points of the cosine function's kth-order derivative is $N_{hp} - 1$, N_{hp} or $N_{hp} + 1$. When fitting a measured signal using a cosine function, it is feasible to judge whether the higher derivative deviates from the physical trend according to whether an abnormal number of extreme points appears.

In summary, there are definite relations between the number of extreme points and the derivative order for the typical process function. Moreover, the pattern of a real physical process could be modeled using a typical function or its combinations. Therefore, the number of extreme points of the higher derivatives could be applied as constraints to fit the measured signal for denoising purpose, which will get noise-reduced signals with higher derivatives conforming to the physical trend.

3. The Extreme-Points-Distribution-Based Neural Network (*EDNN*)

For similar physical processes, the functions modelling the physical quantities varying with time and space have the same form. Additionally, the numbers of the extreme points of those functions and their derivatives are confined in a certain range. Therefore, a novel neural network that introduces the extreme points distribution as constraints was proposed in order to acquire higher derivatives conforming to physical trends. The outline of building the *EDNN* is shown in Figure 1.

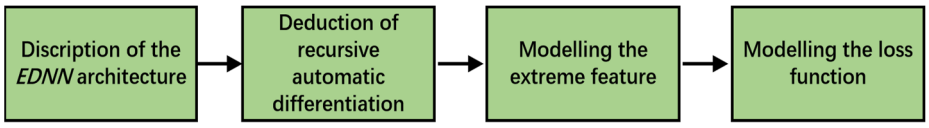

Figure 1. The outline of building the *EDNN*.

3.1. The Architecture of the Extreme-Points-Distribution-Based Neural Network (EDNN)

The extreme-points-distribution-based neural network (*EDNN*) is composed of an input layer, hidden layers, an output layer, an automatic differentiation layer, an extreme points distribution feature layer and a loss function containing extreme points distribution errors. As shown in Figure 2, the *EDNN* contains a sample error calculator (*SEC*) and an extreme points distribution error calculator (*EDEC*). The total loss function of the *EDNN*

is composed of the sample error loss function *LossSp* and the extreme points distribution error loss function *LossEV*.

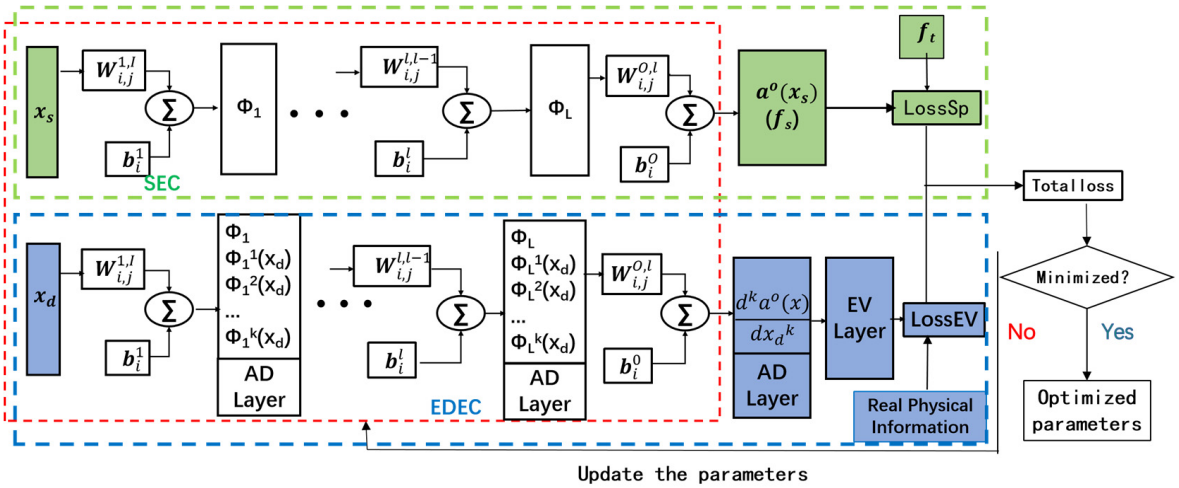

Figure 2. Schematic of the *EDNN*.

The sample error loss function *LossSp* is calculated from the difference between the output f_s of the sample error calculator (*SEC*) and the corresponding target f_t. The extreme points distribution feature error loss function *LossEV* is calculated using the extreme points distribution error calculator (*EDEC*). For each x_d in the measured signal's definition domain, the automatic differentiation layer (*ADLayers*) calculates the 0–(*k*+1)th-order derivatives. Additionally, the extreme points distribution feature layer (*EVLayer*) calculates the number of extreme points of the *0*–*k*th-order derivatives. Then, *LossEV* can be calculated from the difference between the calculated extreme points number and extreme points number of the real physical process.

The total loss function of the *EDNN* (*Loss*) is the weighted summation of *LossSp* and *LossEV*. The *EDNN* is trained, and the weight and the bias are updated until the total loss drops below the stopping criterion or the gradient is less than the stopping criterion.

3.2. Recursive Automatic Differentiation

Calculating the derivatives of the output with respect to the input is necessary for the *EDNN*. A recursive formulation is established for calculating the derivatives. The current layer's derivatives with respect to the input are modeled as a function of the previous layer's derivatives with respect to the input.

3.2.1. Derivatives of Hidden Layers

The derivative of the (*l* + *1*)th hidden layer with respect to the input layer can be formulated as a function of the derivatives of the *l*th hidden layer with respect to the input layer.

The *i*th node's output of the (*l* + *1*)th layer is denoted as $\boldsymbol{a}_i^{l+1}(\boldsymbol{x})$, and its *m*th-order partial derivative with respect to the input layer's *n*th node $\boldsymbol{x}_n^I$ is denoted as $D^m\left(\boldsymbol{a}_i^{l+1}(\boldsymbol{x}), \boldsymbol{x}_n^I\right)$.

According to Faà di Bruno's formula [39,40], $D^m\left(\boldsymbol{a}_i^{l+1}(\boldsymbol{x}), \boldsymbol{x}_n^I\right)$ can be expressed as:

$$\begin{aligned} & D^m\left(\boldsymbol{a}_i^{l+1}(\boldsymbol{x}), \boldsymbol{x}_n^I\right) \\ & = \sum \frac{m!}{b_1!b_2!\cdots b_m!}\left(f_i^{l+1}\right)^{(k)}\left(\sum_{j=1}^{S^l} \boldsymbol{w}_{i,j}^{l+1,l} \cdot \boldsymbol{a}_j^l(\boldsymbol{x}) + \boldsymbol{b}_i^{l+1}\right) \\ & \cdot\left(\frac{\sum_{j=1}^{S^l} \boldsymbol{w}_{i,j}^{l+1,l} \cdot D^1\left(\boldsymbol{a}_j^l(\boldsymbol{x}), \boldsymbol{x}_n^I\right)}{1!}\right)^{b_1} \\ & \cdot\left(\frac{\sum_{j=1}^{S^l} \boldsymbol{w}_{i,j}^{l+1,l} \cdot D^2\left(\boldsymbol{a}_j^l(\boldsymbol{x}), \boldsymbol{x}_n^I\right)}{2!}\right)^{b_2} \cdots \left(\frac{\sum_{j=1}^{S^l} \boldsymbol{w}_{i,j}^{l+1,l} \cdot D^m\left(\boldsymbol{a}_j^l(\boldsymbol{x}), \boldsymbol{x}_n^I\right)}{m!}\right)^{b_m} \end{aligned} \tag{15}$$

where the sum is over all different solutions in non-negative integers $b_1, b_2, \cdots, b_m$ of $b_1 + 2b_2 + \cdots + mb_m = m$ and $k = b_1 + b_2 + \cdots + b_m$; $\left(f_i^{l+1}\right)^{(k)}\left(\sum_{j=1}^{S^l} \boldsymbol{w}_{i,j}^{l+1,l} \cdot \boldsymbol{a}_j^l(\boldsymbol{x}) + \boldsymbol{b}_i^{l+1}\right)$ is the kth-order derivative of the activation function; S^l is the number of nodes in layer l; $\boldsymbol{w}_{i,j}^{l+1,l}$ is the weight of the jth node in the lth layer to the ith node in the $(l+1)$th layer; $\boldsymbol{b}_i^{l+1}$ is the bias of the ith node of the $(l+1)$th layer; $l = 1, \cdots, (L_h - 1)$, L_h is the total number of hidden layers.

When there is no transform in the input layer, the mth-order partial derivative $D^m\left(\boldsymbol{a}_i^1(\boldsymbol{x}), \boldsymbol{x}_n^I\right)$ of the ith node's output $\boldsymbol{a}_i^1(\boldsymbol{x})$ in the first hidden layer with respect to the nth node of the input layer is shown in expression (16).

$$D^m\left(\boldsymbol{a}_i^1(\boldsymbol{x}), \boldsymbol{x}_n^I\right) = \left(f_i^1\right)^{(m)}\left(\sum_{j=1}^{S^I} \boldsymbol{w}_{i,j}^{1,I} \cdot \boldsymbol{x}_j^I + \boldsymbol{b}_i^1\right) \cdot \left(\boldsymbol{w}_{i,n}^{1,I}\right)^m \tag{16}$$

where $\left(f_i^1\right)^{(m)}\left(\sum_{j=1}^{S^I} \boldsymbol{w}_{i,j}^{1,I} \cdot \boldsymbol{x}_j^I + \boldsymbol{b}_i^1\right)$ is the mth-order derivative of the activation function; S^I is the number of nodes of the input layer; $\boldsymbol{w}_{i,j}^{1,I}$ is the weight of the jth node in the input layer to the ith node in the first hidden layer; $\boldsymbol{b}_i^1$ is the bias of the ith node of the first hidden layer.

3.2.2. Derivatives of Output Layers

The partial derivatives of the output layer with respect to the input layer are modeled as the functions of the partial derivatives of the last hidden layer with respect to the input layer. The mth-order partial derivative of the output layer's ith node with respect to the nth node in the input layer is shown in expression (17).

$$D^m\left(\boldsymbol{a}_i^O(\boldsymbol{x}), \boldsymbol{x}_n^I\right) = \sum_{j=1}^{S^{L_h}} \boldsymbol{w}_{i,j}^{O,L_h} \cdot \frac{\partial^m\left[\boldsymbol{a}_j^{L_h}(\boldsymbol{x})\right]}{\partial\left(\boldsymbol{x}_n^I\right)^m} = \sum_{j=1}^{S^{L_h}} \boldsymbol{w}_{i,j}^{O,L_h} \cdot D^m\left(\boldsymbol{a}_i^{L_h}(\boldsymbol{x}), \boldsymbol{x}_n^I\right) \tag{17}$$

where L_h is the total number of hidden layers; S^{L_h} is the number of nodes in the (L_h)th hidden layer; $\boldsymbol{w}_{i,j}^{O,L_h}$ is the weight of the jth node in the (L_h)th hidden layer to the ith node in the output layer.

3.3. Extreme Points Distribution Feature Layer

The role of the extreme points distribution feature layer is calculating the extreme points number. The extreme points of the mth-order partial derivatives $D^m\left(\boldsymbol{a}_i^O(\boldsymbol{x}), \boldsymbol{x}_n^I\right)$ are derived from the $(m+1)$th-order derivative $D^{m+1}\left(\boldsymbol{a}_i^O(\boldsymbol{x}), \boldsymbol{x}_n^I\right)$.

Let:

$$D^{m+1}\left(\boldsymbol{a}_i^O(\boldsymbol{x}), \boldsymbol{x}_n^I\right) = 0 \tag{18}$$

Equation (18) is an equation for $\boldsymbol{x}$.

The solution of Equation (18) is a set:

$$Sol_{m,i,n} = \left\{\boldsymbol{x} \middle| D^{m+1}\left(\boldsymbol{a}_i^O(\boldsymbol{x}), \boldsymbol{x}_n^I\right) = 0\right\} \tag{19}$$

The number of extreme points of $D^m\left(\boldsymbol{a}_i^O(\boldsymbol{x}), \boldsymbol{x}_n^I\right)$ is the rank of the set $Sol_{m,i,n}$:

$$R_{m,i,n} = Rank\left(\boldsymbol{x}\middle| D^{m+1}\left(\boldsymbol{a}_i^O(\boldsymbol{x}), \boldsymbol{x}_n^I\right) = 0\right) \tag{20}$$

where $Rank()$ is the ranking function, $R_{m,i,n}$ is the number of extreme points of $D^m\left(\boldsymbol{a}_i^O(\boldsymbol{x}), \boldsymbol{x}_n^I\right)$ and $R_{m,i,n}$ is defined as the extreme points distribution feature.

3.4. Loss Function Containing Extreme Points Distribution Feature Errors

The number of extreme points of the real physical process is the reference for calculating the extreme points distribution feature error.

If the number of extreme points of derivative $D^m\left(\boldsymbol{a}_i^O(\boldsymbol{x}), \boldsymbol{x}_n^I\right)$ is $R_{m,i,n}$ and the number of extreme points of the real physical process is $TCE_{n,i,m}$, then the extreme points distribution feature error of derivative $D^m\left(\boldsymbol{a}_i^O(\boldsymbol{x}), \boldsymbol{x}_n^I\right)$ is defined as:

$$Er_{m,i,n} = R_{m,i,n} - TCE_{n,i,m} \tag{21}$$

The extreme points distribution feature error loss function $LossEV$ is defined as:

$$LossEV = \sum_{n=1}^{N_I}\sum_{i=1}^{N_o}\sum_{m=0}^{M_d}\left\{\left(Er_{m,i,n}\right)^2\right\} \tag{22}$$

where N_o is the number of nodes in the output layer, N_I is the number of nodes in the input layer and M_d is the highest order of the partial derivatives.

The sample error loss function $LossSp$ is defined as:

$$LossSp = \sum_{s=1}^{N_t}\sum_{i=1}^{N_o}\left[\boldsymbol{t}_{i,s} - \boldsymbol{a}_{i,s}^O(\boldsymbol{x})\right]^2 \tag{23}$$

where $\boldsymbol{t}_{i,s}$ is the ith component of the sth training sample; $\boldsymbol{a}_{i,s}^O(\boldsymbol{x})$ is the ith node's output of the output layer corresponding to the sth training sample; N_t is the number of training samples.

The total loss function of the $EDNN$ is defined as:

$$Loss = S_S{\cdot}LossSp + S_E{\cdot}LossEV \tag{24}$$

where S_S is the weight of the sample error loss function $LossSp$; S_E is the weight of the extreme points distribution feature error loss function $LossEV$.

The optimization criterion is to minimize the total loss function. Additionally, the stopping criterion is when the total loss function is less than 10^{-4} or the gradient is less than 10^{-7}.

4. Application of the *EDNN* in Denoising

The research on signal denoising with $EDNN$ was conducted. Firstly, a single-input, single-output, single-hidden-layer $EDNN$ was realized. Then the $EDNN$ is applied in the denoising of a second-order damped free oscillation signal and an internal combustion engine cylinder pressure trace signal.

4.1. Realization of EDNN

The single-input, single-output $EDNN$ is shown in Figure 3. There are eight nodes in the hidden layer, and the activation function of the hidden layer is a sigmoid function. The output of the sample error calculator (SEC) is:

$$a^o(x) = f\left[\left(\sum_{i=1}^{8} f\left(W_{i,1}^{1,I} * x + b_i^1\right)\right){\cdot}W_{1,i}^{2,1} + b_1^2\right] \tag{25}$$

where $W_{i,1}^{1,I}$ is the weight of the input to the ith node of the first hidden layer; $W_{1,i}^{2,1}$ is the weight of the ith node in the first hidden layer to output; b_i^1 is the bias of the ith node of the first hidden layer; b_1^2 is the bias of the output.

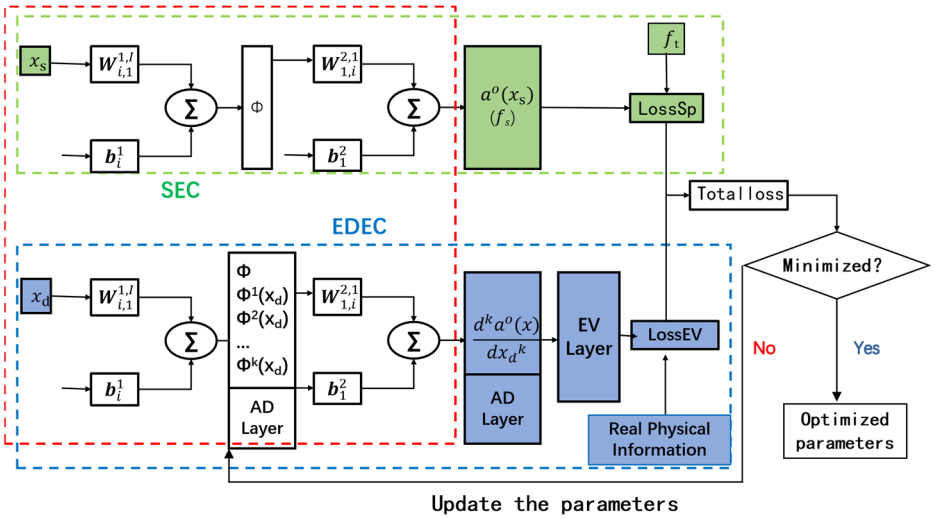

Figure 3. Schematic of the single-input, single-output, single-hidden-layer *EDNN*.

The kth derivative of the output with respect to the input is shown in (26).

$$\frac{d^k a^o(x)}{dx^k} = \sum_{i=1}^{n} f^{(k)}\left(W_{i,1}^{1,I} * x + b_i^1\right) \cdot W_{1,i}^{2,1} * W_{i,1}^{1,I} \tag{26}$$

where $f^{(k)}\left(W_{i,1}^{1,I} * x + b_i^1\right)$ represents the kth-order derivative of the activation function corresponding to the ith node of the hidden layer.

The highest derivative order of the voltage with respect to the time of the second-order damped free oscillation signal is two. Thus, the highest derivative order of the *EDNN* could be set to four, which satisfies the noise reduction demand. Therefore, the maximum value of k in expression (26) is four. Additionally, only the first- through fourth-order derivatives of the activation function are required [41]. The first- through fourth-order derivatives of the activation function are shown in expressions (27)–(30):

$$f'(x) = f(x) - [f(x)]^2 \tag{27}$$

$$f''(x) = f(x) - 3[f(x)]^2 + 2\cdot[f(x)]^3 \tag{28}$$

$$f^{(3)}(x) = f(x) - 7\cdot[f(x)]^2 + 12\cdot[f(x)]^3 - 6\cdot[f(x)]^4 \tag{29}$$

$$f^{(4)}(x) = \left\{f(x) - 15\cdot[f(x)]^2 + 50\cdot[f(x)]^3 - 60\cdot[f(x)]^4 + 24\cdot[f(x)]^5\right\} \tag{30}$$

where $f(x) = \frac{1}{1+e^{-x}}$ is the sigmoid function; $f'(x), f''(x), f^{(3)}(x), f^{(4)}(x)$ are the first, second, third, and fourth derivatives of $f(x)$ with respect to x.

The kth derivative of the output with respect to the input is $\frac{d^k a^o(x)}{dx^k}$. Additionally, the solution of $\frac{d^k a^o(x)}{dx^k} = 0$ is:

$$Sol_{k-1} = \left\{x \middle| \frac{d^k a^o(x)}{dx^k} = 0\right\} \tag{31}$$

The number of extreme points R_{k-1} corresponding to the derivative $\frac{d^{k-1}a^o(x)}{dx^{k-1}}$ is the rank of the set Sol_{k-1}, i.e.:

$$R_{k-1} = Rank\left(x \middle| \frac{d^k a^o(x)}{dx^k} = 0\right) \tag{32}$$

when the number of extreme points of the real physical process corresponding to the derivative $\frac{d^{k-1}a^o(x)}{dx^{k-1}}$ is TCE_{k-1}.

Therefore, the extreme points distribution feature error corresponding to the $(k-1)$th derivative $\frac{d^{k-1}a^o(x)}{dx^{k-1}}$ is:

$$Er_{k-1} = R_{k-1} - TCE_{k-1} \tag{33}$$

where TCE_{k-1} is determined using known physical process information.

The total loss function of the *EDNN* is:

$$Loss = S_S \cdot LossSp + S_E \cdot LossEV = S_S \cdot \sum_{s=1}^{N_t} [t_s - a_s^o(x)]^2 + S_E \cdot \sum_{k=0}^{k=4} (Er_{k-1})^2 \tag{34}$$

where *LossEV* is the extreme points distribution feature error loss function; *LossSp* is the sample error loss function; S_S is the weight of the sample error loss function *LossSp*; S_E is the weight of the extreme points distribution feature error loss function *LossEV*; t_s is target of the *sth* training sample; $a_s^o(x)$ is the output corresponding to the *sth* training sample; N_t is the number of training samples.

4.2. Denoising of Second-Order Damped Free Oscillation Signal via EDNN

4.2.1. Acquisition of Second-Order Damped Free Oscillation Signal

The measured second-order damped free oscillation (*SODFO*) signal has errors, and the effect of the errors on the derivatives is difficult to evaluate. Therefore, a *SODFO* signal is obtained via simulation. High-frequency noise and random noise were superposed on the simulated signal. Then, the performance of the *EDNN* on denoising was studied.

The simulation model of *SODFO* signal is shown in expression (35). Additionally, the simulated signal is a cosine-shape curve with decreasing amplitude (shown in Figure 4a). The simulation model of the high-frequency noise is shown in expression (36). Moreover, the simulated high-frequency noise is a cosine signal with a frequency 10 times the frequency of the *SODFO* signal (shown in Figure 4b). The simulation model of the random noise is shown in expression (37). Further, random noise is a uniformly distributed noise on $[-0.01, 0.01]$ (shown in Figure 4c). The superposition of the higher-order noise and random noise on the *SODFO* signal does not change the amplitude, apparently (shown in Figure 4d).

$$y_c = 10 \cdot e^{(-0.3 \cdot t)} \cdot cos(3 \cdot t) \tag{35}$$

$$h_{noise} = 0.02 \cdot cos(30 \cdot t) \tag{36}$$

$$r_{noise} \sim U(-0.01, 0.01) \tag{37}$$

$$y = y_c + h_{noise} + r_{noise} \tag{38}$$

In expressions (35)–(38), the range of time t is [0.4, 6] seconds; y is the noisy second-order damped free oscillation (*NSODFO*) signal. The time t, signal y and the signal y_c are vectors of length 491.

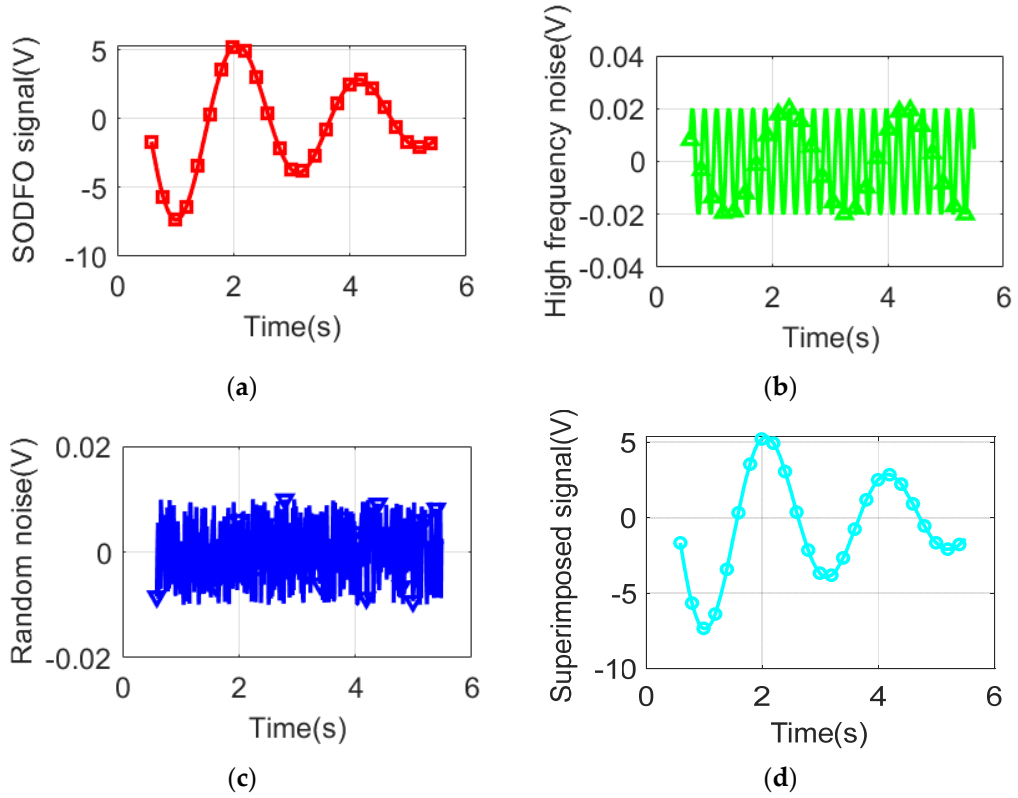

Figure 4. Second-order damped free oscillation signal and noise. (**a**) Simulated second-order damped free oscillation (*SODFO*) signal; (**b**) high-frequency noise; (**c**) random noise; (**d**) noisy second-order damped free oscillation (*NSODFO*) signal.

4.2.2. Comparative Research of *EDNN* with Shallow Neural Network on Denoising of Second-Order Damped Free Oscillation Signal

Comparative research on denoising performance of the *EDNN* was conducted. The *NSODFO* signal processed using the shallow neural network and the *EDNN* were compared. The *SODFO* signal, the noise-removed signal from the shallow neural network (*NRSNN* signal) and the noise-removed signal from the *EDNN* (*NREDNN* signal) are shown in Figure 5.

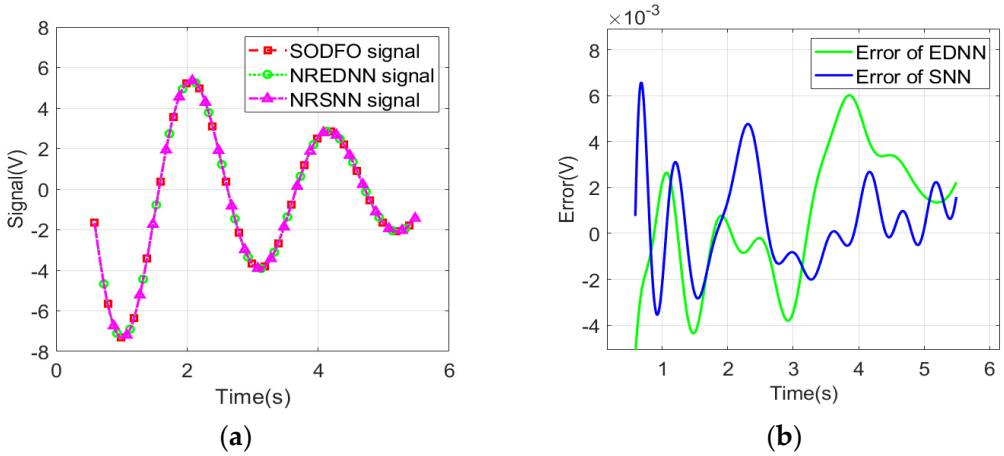

Figure 5. *SODFO* signal, *NRSNN* signal, *NREDNN* signal and the errors. (**a**) *SODFO* signal, *NRSNN* signal, *NREDNN* signal; (**b**) errors between the *NRSNN* signal, *NREDNN* signal and *SODFO* signal.

It can be found in Figure 5a that the *SODFO* signal, *NRSNN* signal and the *NREDNN* signal are well coincident. In Figure 5b, the differences between the denoising signals and

the *SODFO* signal show that the errors of the *NRSNN* signal and the *NREDNN* signal are of the same magnitude. The maximum error of the *NRSNN* signal is 0.09 percent of the maximum amplitude of the *SODFO* signal. The maximum error of the *NREDNN* signal is 0.08 percent of the maximum amplitude of the *SODFO* signal. It indicates that the shallow neural networks and the *EDNN* have similar performance in approximating the amplitude of the signal.

The first- through fourth-order derivatives and the derivative errors are shown in Figure 6.

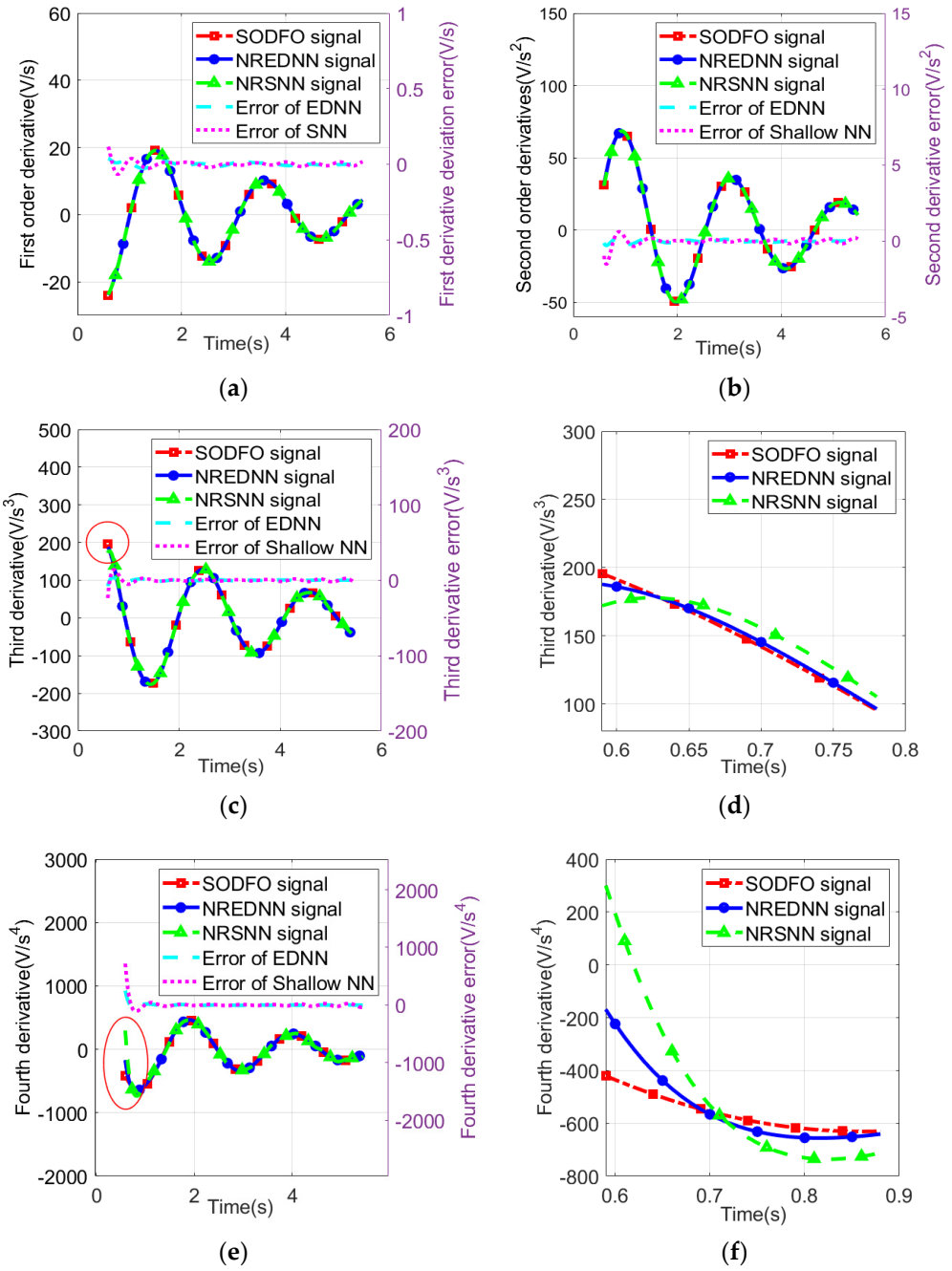

Figure 6. The first- through fourth-order derivatives and the derivative errors. (**a**) First-order derivatives and errors; (**b**) second-order derivatives and errors; (**c**) third-order derivatives and errors; (**d**) magnification of the red circled section in (**c**); (**e**) fourth-order derivatives and errors; (**f**) magnification of the red circled section in (**e**).

Figure 6a is the first-order derivative and the errors. The first-order derivatives of the *NRSNN* signal and the *NREDNN* signal coincide with the first-order derivative of the *SODFO* signal. The maximum error of the first-order derivative of the *NREDNN* signal is 0.16 percent of the maximum amplitude of the first-order derivative of the SODFO signal. The maximum error of the first-order derivative of the *NRSNN* signal is 0.49 percent of the maximum amplitude of the first-order derivative of the *SODFO* signal.

Figure 6b is the second-order derivatives and the errors. The second-order derivatives of the *NRSNN* signal and the *NREDNN* signal coincide with the second-order derivative of the *SODFO* signal. The maximum error of the second-order derivative of the *NREDNN* signal is 0.42 percent of the maximum amplitude of the second-order derivative of the *SODFO* signal. The maximum error of the second-order derivative of the *NRSNN* signal is 2.22 percent of the maximum amplitude of the second-order derivative of the *SODFO* signal.

Figure 6c is the third-order derivatives and the errors. The third-order derivative of the *NREDNN* signal coincides with the third-order derivative of the *SODFO* signal. Additionally, no trend discrepancy occurs in the third-order derivative of the *NREDNN* signal. However, there is a trend discrepancy occurs on the third-order derivative of the *NRSNN* signal (Figure 6d). The maximum error of the third-order derivative of the *NREDNN* signal is 4.02 percent of the maximum amplitude of the third-order derivative of the *SODFO* signal. The maximum error of the third-order derivative of the *NRSNN* signal is 12.05 percent of the maximum amplitude of the third-order derivative of the *SODFO* signal.

Figure 6e is the fourth-order derivatives and the errors. The fourth-order derivatives of the *NRSNN* signal and the *NREDNN* signal coincide with the fourth-order derivative of the *SODFO* signal. The maximum error of the fourth-order derivative of the *NREDNN* signal is 39.92 percent of the maximum amplitude of the fourth-order derivative of the *SODFO* signal. The maximum error of the fourth-order derivative of the *NRSNN* signal is 114.24 percent of the maximum amplitude of the fourth-order derivative of the *SODFO* signal. The error of *NREDNN* signal is less than that of the *NRSNN* signal (Figure 6f).

The standard deviation of the first- through fourth-order derivatives are shown in Table 1. The standard deviation error between the derivatives of the *SODFO* signal and that of the *NREDNN* signal is less than 62.5 percent of the standard deviation errors between the derivatives of the *SODFO* signal and the *NRSNN* signal.

Table 1. The standard deviation of the derivatives.

	1st Derivative	2nd Derivative	3rd Derivative	4th Derivative
SNN_18 neural network	0.016	0.215	2.642	28.692
EDNN neural network	0.010	0.074	0.751	8.414

In summary, the noise reduction performance of the *NREDNN* is better than that of the shallow neural networks. Additionally, no trend discrepancy occurred in the first- through fourth-order derivatives of the *NREDNN* signal. Those are the advantages of the *EDNN*. However, the *EDNN* requires more memory to store the derivatives and more computational time to calculate the derivative and the extreme feature error.

4.3. Denoising of the Cylinder Pressure Signal with EDNN

The internal combustion engine cylinder pressure trace is the variation of cylinder pressure with crank angle. Generally, the cylinder pressure trace is recorded using a piezoelectric high-temperature pressure sensor and smoothed via cyclic averaging [42]. The cyclically averaged cylinder pressure trace can satisfy the demands of calculation of indicated mean effective pressure. However, when the analysis involves derivatives

beyond the second order, the noise often leads to deviation from the physical trend. The denoising performance of the *EDNN* on cylinder pressure trace was studied.

The cylinder signal is the in-cylinder pressure which varies with the crank angle. The pressure signal is between the −120 degree CA and 120 degree CA, and the sample interval is 0.1 degrees CA. Additionally, the cylinder pressure signal is a vector of length 2400 whose maximum value is 1.6946 $\times$ 10^6 Pa.

The single-input, single-hidden-layer, single-output *EDNN* with eight nodes in the hidden layer was applied to reduce the noise of the cylinder pressure trace. Moreover, the noise reduction results were compared with that from a shallow neural network with 8 nodes (*SNN_8*), a shallow neural network with 18 nodes (*SNN_18*) and the smoothing splines that are traditionally adopted in smoothing cylinder pressure trace.

The noise reduction results are shown in Figure 7. The noise-reduced signals of the four techniques are all consistent with the measured cylinder pressure (Figure 7a). Additionally, the errors between the noise-reduced signal and the measured signal in Figure 7b are all less than 400 Pa, and the relative error is less than 1.2 $\times$ 10^{-6}%, which satisfies the accuracy requirements.

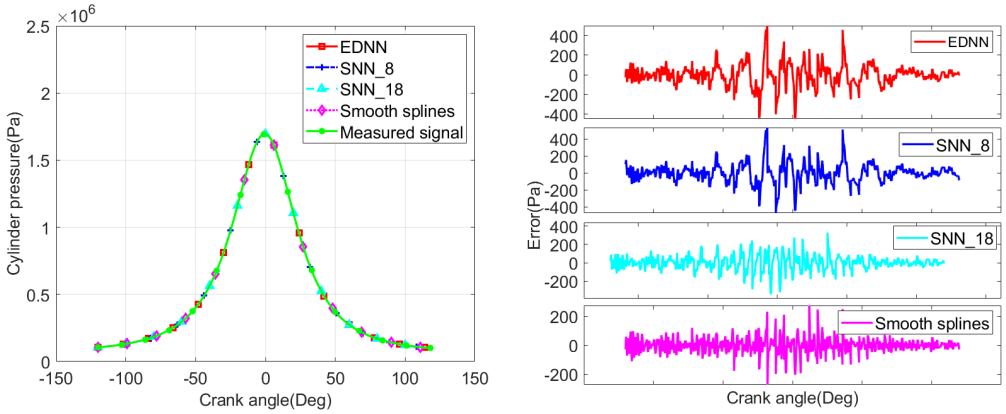

Figure 7. Cylinder pressures and errors. (**a**) Measured cylinder pressure signal and noise-reduced signals; (**b**) the errors between the noise-reduced signals and the measured signal.

Figure 8 shows the first- through fourth-order derivatives. It can be seen in Figure 8a that the first-order derivatives of the noise-reduced signal from the four techniques are all consistent with each other, and there are no fluctuations deviating from the physical trend. The first-order derivative of the measured signal shows a small fluctuation near the zero-degree crank angle, which deviates from the physical trend.

It can be seen in Figure 8b that the second-order derivative of the noise-reduced signal from the *EDNN* and the *SNN_18* neural network is consistent with real trend and there is no fluctuations deviating from the physical trend. The second-order derivatives of the noise-reduced signal from the *SNN_8* neural network and the smoothing splines showed small fluctuations. Further, the second-order derivative of the measured signal shows frequent fluctuations deviating from the physical trend in the whole crank angle range.

It can be seen in Figure 8c that there are no fluctuations deviating from the physical trend occur on the third-order derivative of the noise-reduced signal, obtained from the *EDNN*. There are small fluctuations on the third-order derivative of the noise-reduced signal from the *SNN_18* neural network. In addition, there are obvious fluctuations on the third-order derivative of the noise-reduced signal from the *SNN_8* neural network. There are frequent fluctuations on the third-order derivative of the noise-reduced signal, obtained from the smoothing spline. Additionally, the third-order derivative of the measured signal is submerged in the noise.

It can be seen in Figure 8d that there are no fluctuations deviating from the physical trend on the fourth-order derivative of the noise-reduced signal, obtained from the *EDNN*.

There are obvious fluctuations on the fourth-order derivative of the noise-reduced signal from the *SNN_18* neural network and the *SNN_8* neural network. Additionally, the fourth-order derivative of the measured signal and the noise-reduced signal obtained from the smoothing splines are submerged in the noise.

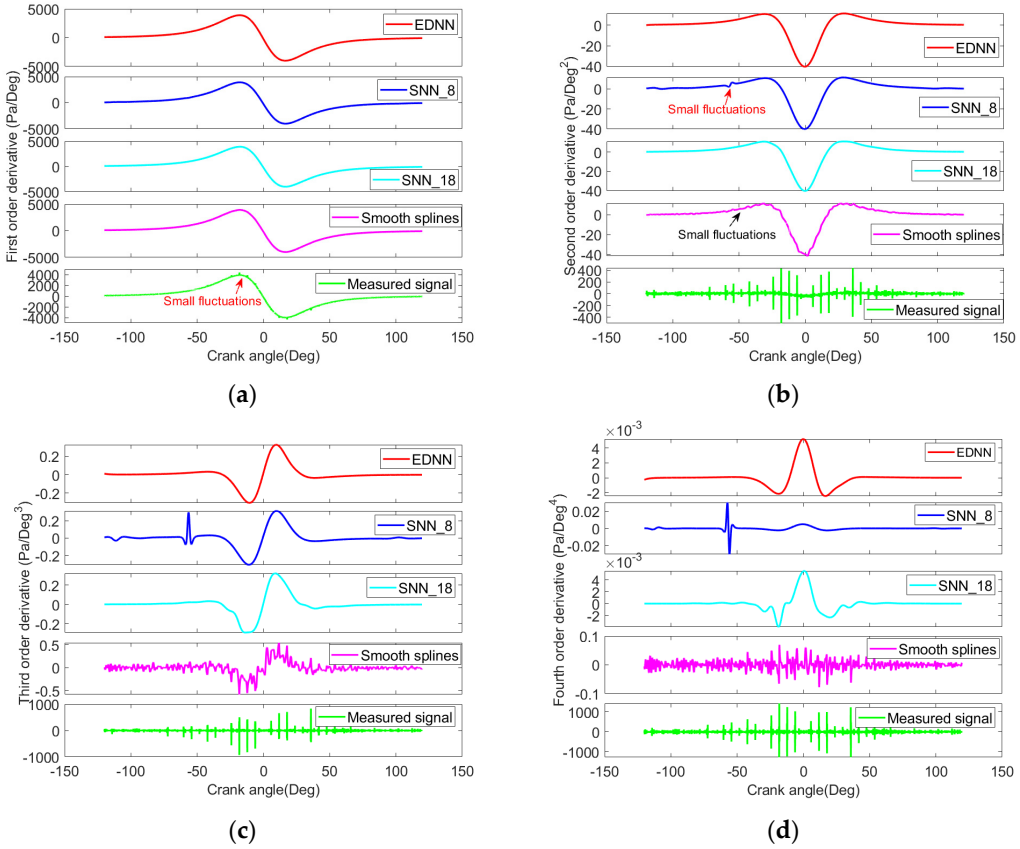

Figure 8. The first- through fourth-order derivatives. (**a**) First-order derivatives; (**b**) second-order derivatives; (**c**) third-order derivatives; (**d**) fourth-order derivatives.

In summary, the noise reduction performance of the *EDNN* is better than that of the *SNN_8* neural network, the *SNN_18* neural network and the smoothing splines. Further, there is no trend discrepancy in the first- through fourth-order derivatives of the noise-reduced cylinder pressure signal from the *EDNN*.

The *EDNN* could derive higher derivatives consistent with the real physical process in the absence of the process' detailed mathematical model. However, the *EDNN* needs more computational resources due to the recording and calculation of the higher derivatives.

5. Conclusions

The effect of the error on the higher derivatives of the measured signal was analyzed. The feasibility of using the extreme points distribution as constraints in data fitting was studied. Then, the extreme-points-distribution-based neural network (*EDNN*) was proposed. Finally, the superiority of the *EDNN* on signal denoising was verified. The details are as follows:

1. The error's deviation effect on the higher derivatives of the measured signal was analyzed and a possible way of applying the extreme points distribution as constraints on data fitting was studied.

A mathematical model was established for the analysis of the error's effect on higher derivatives. It was found that even though the Taylor series of the noise only has the kth-order term and the coefficients of other terms are zero, the kth-order derivative of the measured signal could deviate from the real physical process greatly as long as the

*k*th-order derivative of the noise is large enough. The necessity of finding a way to fit data with higher derivatives compatible to real process trends was clarified.

The higher derivative's extreme points distribution of typical process functions were analyzed, and the pattern of the extreme points distribution of higher derivatives was investigated. The pattern provides a theoretical basis for applying the extreme points distribution as a constraint in denoising with data fitting techniques.

2. The extreme points distribution pattern was adopted as a constraint and the *EDNN* was established. The *EDNN* consists of an input layer, hidden layers, an output layer, an automatic differentiation layer and an extreme points distribution feature layer. A recursive formulation was established for calculating the derivatives. Additionally, a novel loss function, embedded with the extreme feature error, was proposed.
3. The effectiveness of the *EDNN* on signal denoising was verified. The proposed *EDNN* was applied to reduce the noise in the second-order damped free oscillation signal and the internal combustion engine cylinder pressure signal. Compared with shallow neural networks and smoothing splines, the *EDNN* could obtain higher derivatives that are consistent with the real physical process in the absence of a detailed mathematical model. Therefore, data fitting for higher derivatives conforming to real physical process trends could be realized with *EDNN*, which provides a novel approach for analyzing physical processes with higher derivatives. Additionally, it could be used in understanding the real process with higher derivatives.

In summary, the advantage of the *EDNN* is that it could fit the measured signal with higher derivatives compatible to real physical process trends, which provides a novel tool to mine information of the real physical process through higher derivatives. However, it needs more computational resources and needs the knowledge of extreme distribution of the real process. The aim of future research is to study the computational efficiency of the *EDNN* and apply it in other engineering fields.

Author Contributions: Conceptualization, X.W., F.M. and Y.G.; methodology, X.W., F.M., J.L. and Y.G.; software, X.W. and F.M.; validation, X.W., F.M. and J.L.; formal analysis, X.W.; investigation, X.W. and F.M.; resources, X.W.; data curation, X.W., F.M. and J.L.; writing—original draft preparation, X.W., F.M. and J.L.; writing—review and editing, X.W., F.M., J.L., Y.G. and C.Z.; visualization, F.M., J.L. and X.W.; supervision, X.W.; project administration, X.W.; funding acquisition, X.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Ganz, A.; Noui, K. Reconsidering the Ostrogradsky theorem: Higher-derivatives Lagrangians, ghosts and degeneracy. *Class. Quantum Gravity* **2021**, *38*, 075005. [CrossRef]
2. Visser, M. Jerk, snap, and the cosmological equation of state. *Class. Quantum Gravity* **2003**, *21*, 2603–2615. [CrossRef]
3. Eager, D.; Pendrill, A.M.; Reistad, N. Beyond velocity and acceleration: Jerk, snap and higher derivatives. *Eur. J. Phys.* **2016**, *37*, 065008. [CrossRef]
4. Alwan, Z.M. Solution Non Linear Partial Differential Equations By ZMA Decomposition Method. *WSEAS Trans. Math.* **2021**, *20*, 712–716. [CrossRef]
5. Al-Habahbeh, A. Numerical Solution of a System of Fractional Ordinary Differential Equations by a Modified Variational Iteration Procedure. *WSEAS Trans. Math.* **2022**, *21*, 309–318. [CrossRef]
6. Faris, H.S.; Butris, R.N. Existence, Uniqueness, and Stability of Solutions of Systems of Complex Integrodifferential Equations on Complex Planes. *WSEAS Trans. Math.* **2022**, *21*, 90–97. [CrossRef]
7. Zhang, S.; Chen, Z.; Zhang, W.; Xia, M. Research on dynamic stiffness with the high-order fractional derivative model for rubber bushing. *Proc. Inst. Mech. Eng. Part D J. Automob. Eng.* **2022**, *237*, 09544070221079504. [CrossRef]

8. Abouelregal, A.E. A novel generalized thermoelasticity with higher-order time-derivatives and three-phase lags. *Multidiscip. Model. Mater. Struct.* **2020**, *16*, 689–711. [CrossRef]
9. Brusa, A.; Corti, E.; Rossi, A.; Moro, D. Enhancement of Heavy-Duty Engines Performance and Reliability Using Cylinder Pressure Information. *Energies* **2023**, *16*, 1193. [CrossRef]
10. Chen, Y.; Cao, R.; Chen, J.; Liu, L.; Matsushita, B. A practical approach to reconstruct high-quality Landsat NDVI time-series data by gap filling and the Savitzky–Golay filter. *ISPRS J. Photogramm. Remote Sens.* **2021**, *180*, 174–190. [CrossRef]
11. Zuo, B.; Cheng, J.; Zhang, Z. Degradation prediction model for proton exchange membrane fuel cells based on long short-term memory neural network and Savitzky-Golay filter. *Int. J. Hydrogen Energy* **2021**, *46*, 15928–15937. [CrossRef]
12. Tong, Y.; Wang, L.; Zhang, W.-Z.; Zhu, M.-D.; Qin, X.; Jiang, M.; Rong, X.; Du, J. A high performance fast-Fourier-transform spectrum analyzer for measuring spin noise spectrums. *Chin. Phys. B* **2020**, *29*, 090704. [CrossRef]
13. Wang, K.; Wen, H.; Li, G. Accurate frequency estimation by using three-point interpolated discrete fourier transform based on rectangular window. *IEEE Trans. Ind. Inform.* **2020**, *17*, 73–81. [CrossRef]
14. Jalayer, M.; Orsenigo, C.; Vercellis, C. Fault detection and diagnosis for rotating machinery: A model based on convolutional LSTM, Fast Fourier and continuous wavelet transforms. *Comput. Ind.* **2021**, *125*, 103378. [CrossRef]
15. Ghimire, S.; Deo, R.C.; Raj, N.; Mi, J. Wavelet-based 3-phase hybrid SVR model trained with satellite-derived predictors, particle swarm optimization and maximum overlap discrete wavelet transform for solar radiation prediction. *Renew. Sustain. Energy Rev.* **2019**, *113*, 109247. [CrossRef]
16. Raissi, M.; Perdikaris, P.; Karniadakis, G.M. A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *J. Comput. Phys.* **2019**, *378*, 686–707. [CrossRef]
17. Yu, J.; Lu, L.; Meng, X.; Karniadakis, G.E. Gradient-enhanced physics-informed neural networks for forward and inverse PDE problems. *Comput. Methods Appl. Mech. Eng.* **2022**, *393*, 114823. [CrossRef]
18. Cuomo, S.; Di Cola, V.S.; Giampaolo, F.; Rozza, G.; Raissi, M.; Piccialli, F. Scientific machine learning through physics–informed neural networks: Where we are and what's next. *J. Sci. Comput.* **2022**, *92*, 88. [CrossRef]
19. Margossian, C.C. A review of automatic differentiation and its efficient implementation. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* **2019**, *9*, e1305. [CrossRef]
20. Baydin, A.G.; Pearlmutter, B.A.; Radul, A.A.; Siskind, J.M. Automatic differentiation in machine learning: A survey. *J. Mach. Learn. Res.* **2018**, *18*, 5595–5637.
21. Novikov, A.; Rakhuba, M.; Oseledets, I. Automatic differentiation for Riemannian optimization on low-rank matrix and tensor-train manifolds. *SIAM J. Sci. Comput.* **2022**, *44*, A843–A869. [CrossRef]
22. Pombo, I.; Sarmento, L. Automatic differentiation as an effective tool in Electrical Impedance Tomography. *arXiv* **2023**, arXiv:2301.11410.
23. Kim, J.; Lee, K.; Lee, D.; Jhin, S.Y.; Park, N. DPM: A novel training method for physics-informed neural networks in extrapolation. In Proceedings of the AAAI Conference on Artificial Intelligence, Virtual, 22 February–1 March 2022; Volume 35, pp. 8146–8154.
24. Linka, K.; Schäfer, A.; Meng, X.; Zou, Z.; Karniadakis, G.E.; Kuhl, E. Bayesian Physics Informed Neural Networks for real-world nonlinear dynamical systems. *Comput. Methods Appl. Mech. Eng.* **2022**, *402*, 115346. [CrossRef]
25. Zhu, Y.; Zabaras, N.; Koutsourelakis, P.-S.; Perdikaris, P. Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data. *J. Comput. Phys.* **2019**, *394*, 56–81. [CrossRef]
26. Sengupta, S.; Basak, S.; Saikia, P.; Paul, S.; Tsalavoutis, V.; Atiah, F.; Ravi, V.; Peters, A. A review of deep learning with special emphasis on architectures, applications and recent trends. *Knowl. Based Syst.* **2020**, *194*, 105596. [CrossRef]
27. Schiassi, E.; Furfaro, R.; Leake, C.; De Florio, M.; Johnston, H.; Mortari, D. Extreme theory of functional connections: A fast physics-informed neural network method for solving ordinary and partial differential equations. *Neurocomputing* **2021**, *457*, 334–356. [CrossRef]
28. Dwivedi, V.; Srinivasan, B. Physics Informed Extreme Learning Machine (PIELM)—A rapid method for the numerical solution of partial differential equations. *Neurocomputing* **2020**, *391*, 96–118. [CrossRef]
29. Ciulla, G.; D'Amico, A.; Di Dio, V.; Lo Brano, V. Modelling and analysis of real-world wind turbine power curves: Assessing deviations from nominal curve by neural networks. *Renew. Energy* **2019**, *140*, 477–492. [CrossRef]
30. Tong, Y.; Yu, L.; Li, S.; Liu, J.; Qin, H.; Li, W. Polynomial fitting algorithm based on neural network. *ASP Trans. Pattern Recognit. Intell. Syst.* **2021**, *1*, 32–39. [CrossRef]
31. Yarotsky, D. Error bounds for approximations with deep ReLU networks. *Neural Netw.* **2017**, *94*, 103–114. [CrossRef] [PubMed]
32. Braga-Neto, U. Characteristics-Informed Neural Networks for Forward and Inverse Hyperbolic Problems. *arXiv* **2022**, arXiv:2212.14012.
33. Yang, Y.; Perdikaris, P. Adversarial uncertainty quantification in physics-informed neural networks. *J. Comput. Phys.* **2019**, *394*, 136–152. [CrossRef]
34. Yang, L.; Meng, X.; Karniadakis, G.E. B-PINNs: Bayesian physics-informed neural networks for forward and inverse PDE problems with noisy data. *J. Comput. Phys.* **2021**, *425*, 109913. [CrossRef]
35. Harrach, B.; Pohjola, V.; Salo, M. Monotonicity and local uniqueness for the Helmholtz equation. *Anal. PDE* **2019**, *12*, 1741–1771. [CrossRef]
36. Pascoe, J.E.; Tully-Doyle, R. The royal road to automatic noncommutative real analyticity, monotonicity, and convexity. *Adv. Math.* **2022**, *407*, 108548. [CrossRef]

37. Sadat, A.; Joye, I.J. Peak fitting applied to fourier transform infrared and Raman spectroscopic analysis of proteins. *Appl. Sci.* **2020**, *10*, 5918. [CrossRef]
38. Kara, M. An analytical expression for arbitrary derivatives of Gaussian functions exp(ax2). *Int. J. Phys. Sci.* **2009**, *4*, 247–249.
39. Johnson, W.P. The Curious History of Faà di Bruno's Formula. *Am. Math. Mon.* **2002**, *109*, 217–234.
40. Frabetti, A.; Mancho, D. Five interpretations of Faà di Bruno's formula. *arXiv* **2014**, arXiv:1402.5551.
41. Mellinger, D.; Kumar, V. Minimum snap trajectory generation and control for quadrotors. In Proceedings of the 2011 IEEE International Conference on Robotics and Automation, Shanghai, China, 9–13 May 2011; pp. 2520–2525.
42. Pulpeiro, G.J.; Hall, C.M.; Kolodziej, C.P. Determination of a most representative cycle from cylinder pressure ensembles via statistical method using distribution skewness. *Int. J. Engine Res.* **2023**, *24*, 720–737. [CrossRef]