IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received June 8, 2020, accepted June 21, 2020, date of publication June 25, 2020, date of current version July 8, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3004886

Adaptive Artificial Neural Network Surrogate Model of Nonlinear Hydraulic Adjustable Damper for Automotive Semi-Active Suspension System

JINGLIANG LIN[1], HAIYAN LI[1], YUNBAO HUANG[1,2], ZEYING HUANG[1], AND ZHIQIAN LUO[3]

[1]College of Mechanical and Electrical Engineering, Guangdong University of Technology, Guangzhou 510006, China
[2]Foshan Chuwei Technology Company Ltd., Foshan 528200, China
[3]Engineering Institute, Guangzhou Automobile Group Company Ltd., Guangzhou 510640, China

Corresponding authors: Haiyan Li (cathylhy@gdut.edu.cn) and Yunbao Huang (huangyb@gdut.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 51775116, Grant 51975125, Grant 51375185, and Grant 51405177, in part by the joint fund of the National Natural Science Foundation Committee of China and China Academy of Engineering Physics (NSAF) under Grant U1430124, and in part by the Innovation and Entrepreneurship Team Program of CNC Equipment Cooperative Innovation Institute of Foshan Nanhai and Guangdong University of Technology.

ABSTRACT Artificial neural network surrogate models are often used in the design optimization of the automotive semi-active suspension system. To realize the desired damping force, a surrogate model needs to be constructed to approximate the regulating mechanism of the hydraulic adjustable damper. However, very few of the existing studies discuss how to guarantee the modeling accuracy. To this end, this work constructs a novel surrogate model by using radial basis function neural network. Meanwhile, an adaptive modeling method based on modified hyperband and trust-region-based mode pursuing sampling is presented. Concretely, modified hyperband is used to fast select a seed model by early-stopping and dynamic resource allocation. Mode pursuing sampling is then performed in the neighborhood of the seed model, to systematically generate more sample points while statistically covering the entire neighborhood (i.e., trust region). In particular, in the mode pursuing sampling procedure, quadratic regression is performed around the current optimum as the second detection. Moreover, as the position or size of the trust region changes, the sampling and detection process iterate until the accuracy of the model is no longer improved. To avoid falling into the local optimum, the seed model selection and mode pursuing sampling process iterate until the termination criterion is met. The experimental results show that compared with the benchmarks, the modeling accuracy of hydraulic adjustable damper can be improved by up to 48%, and the iteration resources can be reduced by up to 84%.

INDEX TERMS Semi-active suspension systems, hydraulic adjustable damper, artificial neural network, hyperparameter optimization.

I. INTRODUCTION

The performance of the vehicle is affected by its suspension system [1], [2], such as ride comfort and handling stability. There are three main types of suspension systems [3]–[6]: passive suspension, semi-active suspension (SAS), and active suspension (AS). In general, the passive suspension is difficult to meet high performance requirements in actual use, because its spring stiffness and damping coefficient are not adjustable. Therefore, the use of controllable suspension systems (i.e., SAS and AS) is an inevitable choice for high-performance vehicles [7]–[12]. Compared with the active type, the spring stiffness and damping coefficient of SAS are adjusted according to the optimized parameters of springs and shock absorbers under various conditions stored inside the computer, resulting in low energy consumption,

The associate editor coordinating the review of this manuscript and approving it for publication was Lefei Zhang.

reliable performance, and low cost [13]–[15]. Hence, this research focuses on the SAS system.

The controllable SAS system usually means that the damper is adjustable. Among them, the monotube hydraulic adjustable damper (HAD) using the hydraulic flow regulation method is the most popular in the automotive industry [16]–[18]. To facilitate the practical application of the monotube HAD, its regulating mechanism should be modeled accurately, which means that the characteristic of the monotube HAD should be available in the design stage. With regards to this, the analytical model and thermal effect equations of the monotube HAD were established in Ref. [19], and their feasibility were verified by the integral shock absorber testing (ISAT) system. On this basis, HAD data was further collected from the ISAT system to construct a gray neural network (GNN) surrogate model of the unknown regulating mechanism. To ensure the accuracy and decrease the computational cost, a further research using fuzzy neural network (FNN) has been conducted in Ref. [6], and performance improvement in modeling was obtained.

In theory, artificial neural network (ANN) has the capability of approximating any complex and non-linear function arbitrarily well, and can be applied in many fields as a prediction model or surrogate model, such as performance prediction and decision optimization [19]–[22], fault diagnosis and fault tolerant control [23]–[25], and image recognition and classification [26]–[28]. However, the approximation performance of different ANNs in engineering applications varies [6], [22]. In the existing researches, only a few ANNs are used to solve the HAD problem [6], [19]. Therefore, a new neural network should be investigated in order to accurately realize the desired damping force. To this end, this work constructs a novel surrogate model by using radial basis function neural network (RBFNN) [21], [22]. RBFNN is employed as it has fast convergence speed and powerful anti-noise and repair capabilities, which is more in line with the requirements of SAS systems.

For the practical application of ANN surrogates, the hyperparameters need to be intentionally tuned to accurately approximate the training data. Hyperparameter optimization is a time-consuming task. To reduce the computational cost, in the existing HAD researches [6], [12], only a few of hyperparameters have been optimized, which may lead to the failure of obtaining a high-precision model. To solve this problem, an adaptive modeling method based on hyperparameter optimization algorithm is presented in this work. The ''adaptive'' refers to automatic selection, relative to manual [6] and semi-adaptive [12] methods.

Hyperparameter optimization algorithm has been widely used in the field of deep learning, such as random search (RS) [28], [29], Bayesian optimization (BO) [30]–[35] and hyperband [36]. However, RS cannot guarantee the global optimum, and is inefficient in high-dimensional problems; BO that based on probabilistic model requires calculation of complex statistics. As the number of evaluated points increases, the calculation cost increases sharply, resulting in low efficiency; hyperband that integrates resource dynamic allocation and early stopping strategies into RS to quickly eliminate the hyperparameter configurations with early poor performance, which has been proved to obtain better results than BO. However, its performance is limited by the RS part. Therefore, to achieve efficient adaptive modeling, a new hyperparameter method which combines modified hyperband (MH) with mode pursuing sampling (MPS) [37] through a trust region (TR) strategy is proposed, and is referred to as MH-TRMPS.

In MH-TRMPS, TR indicates a region around the current optimum, and within which MPS is performed to obtain minimizer of this region. Different from existing gradient driven trust-region-based optimization methods [38]–[41], MPS search is independent in MH-TRMPS, and there is no need to calculate the step size or direction. Concretely, MH is used to fast select the current optimum. Meanwhile, a hypercube around the optimum is constructed and defined as TR. MPS is then performed in the TR to pursue a more accurate model. According to the performance of the MPS search during previous iterations, the size of TR is either remained the same or reduced by a preset ratio, but its center always falls on the optimum. If the candidate solution does not produce a sufficient increase in the modeling accuracy after multiple reduction of the size of TR, MH search will be conducted again to determine whether a new TR needs to be reconstructed.

The contributions of this work can be summarized as follows:

(1) An efficient adaptive ANN surrogate modeling method based on MH-TRMPS is presented for the HAD problem. Numerical examples confirm that the proposed approach can obtain better performance than manual method, semi-adaptive method, and other adaptive methods. To the best of our knowledge, this is the first investigation of the adaptive ANN-surrogate for HAD problem.

(2) A novel surrogate model is constructed for the HAD problem by using RBFNN. Through the application on the simulation data of HAD, the modeling accuracy of RBFNN is better than those of MLP and FNN. Specifically, the application of RBFNN has not been used for HAD problem in the past.

The rest of this paper is organized as follows. Section 2 shows the related materials of HAD. Section 3 introduces the MLP, FNN, and RBFNN models of nonlinear HAD. In Section 4, MH-TRMPS method is presented in detail. Section 5 introduces the numerical experiments. The conclusion is given in Section 6.

II. REGULATION MECHANISM OF THE NONLINEAR HAD

Fig.1 shows a commercial SAS system with HAD. (a) is the basic setup of a SAS system [42], and (b) is the structural diagram of a nonlinear HAD. In Fig. 1(b), the outermost side is the damper cylinder block, and the upper side is the damper end cap. Above and below the piston are oil reservoirs, respectively referred to as the upper hydraulic oil

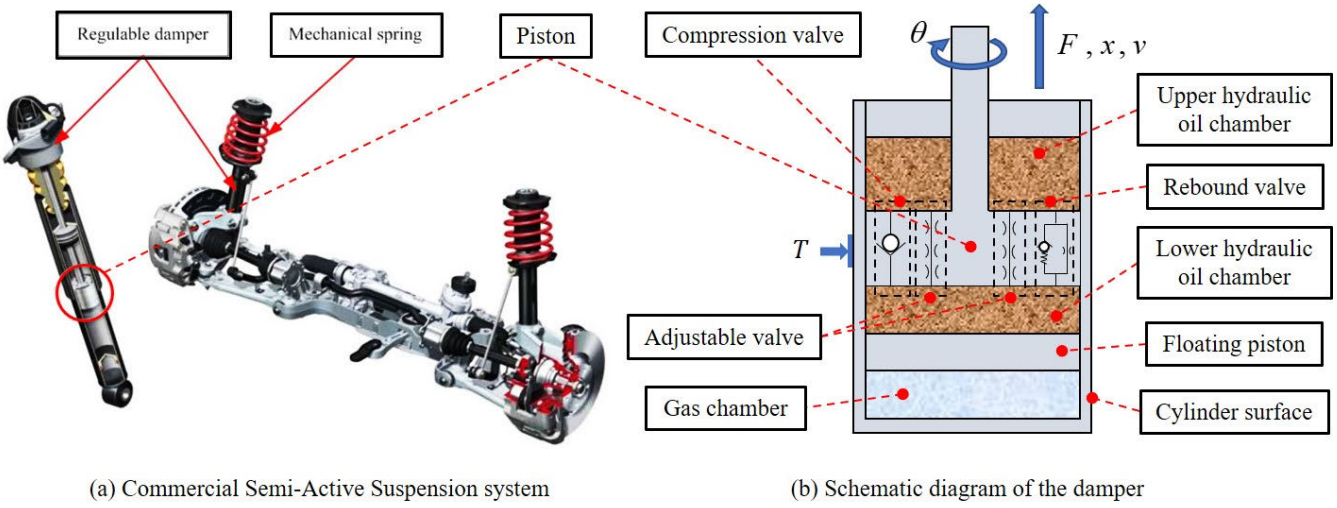

FIGURE 1. SAS system with the nonlinear HAD.

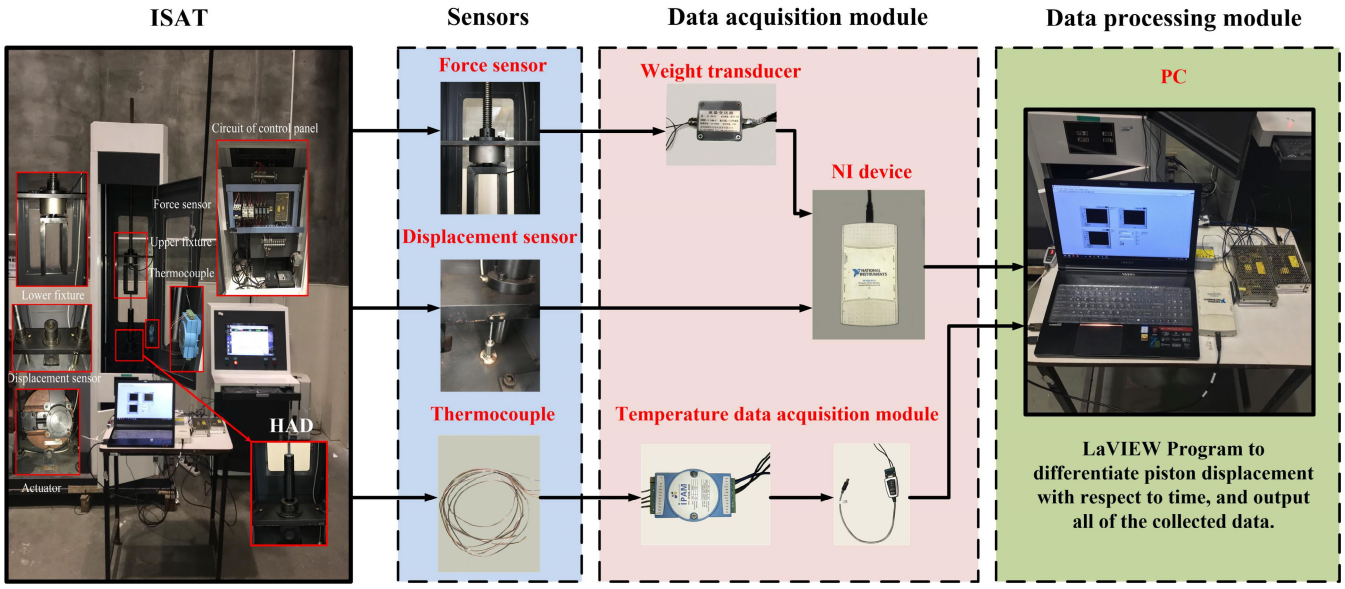

FIGURE 2. The process of data collection.

chamber and the lower hydraulic oil chamber. At the bottom of the damper is a gas chamber, and between the gas chamber and the lower hydraulic oil chamber is a floating piston.

The working principle of nonlinear HAD consists of two parts: the compression stroke and the rebound stroke. For the compression stroke, as the piston moves downward, the hydraulic oil in the lower hydraulic oil chamber flows into the upper hydraulic oil chamber through the adjustable valve inside the piston, while pushing the floating piston downward to compress the gas in the gas chamber. For the rebound stroke, as the piston moves upward, the hydraulic oil in the upper hydraulic oil chamber flows into the lower hydraulic oil chamber through the rebound valve in the piston assembly and the adjustable valve inside the piston. In order to compensate for the volume difference caused by the piston leaving the cylinder, the floating piston moves upward.

The process of HAD data collection is shown in Fig. 2. From the ISAT, each group of data contains five variables: piston displacement x (m), cylinder surface temperature T (°C), motor angle θ (degree), damping force F (N), and piston speed v (m/s). The first four variables are taken directly from the tester, and the piston speed v is obtained in the LabView program by the differential of the piston displacement versus time. In this work, the hydraulic oil temperature is replaced by the cylinder surface temperature T, because the hydraulic oil is sealed in the cylinder and it is difficult to measure its temperature. Then, the regulating mechanism of damping force for nonlinear HAD can be expressed as a function of

the variables as follows:

$$F = f\,(v, x, \theta, T)\,, \tag{1}$$

The values of the data collected directly from the ISAT differ widely, for example, the force range from −1000 to 3500, and the displacement range from −45 to 45. Consequently, the method in Ref. [6] is used to normalize the data and is defined as follows:

$$\overline{x_i} = \frac{2(x_i - x_{i\_min})}{x_{i\_max} - x_{i\_min}} - 1, \tag{2}$$

where x_{i_min} and x_{i_max} represent the maximum and minimum values of x_i, respectively, and $\overline{x_i}$ are the variables after normalization.

III. ANN MODELS OF THE NONLINEAR HAD

In this section, three ANN surrogate models, i.e., MLP, FNN, and RBFNN, are designed for the nonlinear HAD. Besides, the learning algorithm for training the models is also introduced.

A. MODEL DESIGN

1) MULTILAYER PERCEPTRON (MLP) MODEL

MLP is a fully connected network, which has been widely used due to its low prior knowledge dependency. As shown in Fig. 3, the general structure of the MLP model usually includes an input layer, an output layer, and n hidden layers. Obviously, to construct the MLP model, the number of hidden layers n and the number of neurons m_l in the hidden layers need to be determined first, where $1 \le l \le n$. n and m_l are the so-called structure hyperparameters of MLP, w_{ij}^l, b^l are the parameters of MLP. Generally, the larger the training data set is, the larger n and m_l should be chosen to ensure an accurate model.

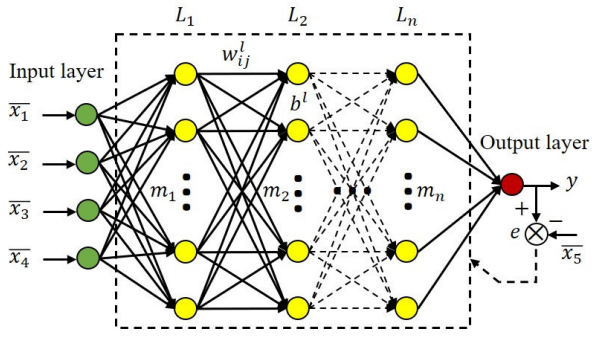

FIGURE 3. General structure of MLP model.

In the MLP model, the same activation function is usually used in hidden neurons to map the feature of the training data. In this work, the generally rectified linear unit (ReLU) [43] is chosen as activation function, because it does not have the problem of gradient vanishing when the input is positive. Moreover, its calculation speed is much faster than Sigmod and Tanh. The ReLU function is defined as:

$$f\left(y_j^l\right) = max\left(0, y_j^l\right), \tag{3}$$

and y_j^l is calculated by

$$y_j^l = \sum_{i=1}^{m_{l-1}} w_{ji}^l x_i^{l-1} + b^l, \tag{4}$$

where x_i^{l-1} is the ith output of the $(l-1)$th layer, y_j^l is the jth output of the lth layer, w_{ji}^l is the weight connecting the jth neuron of the lth layer and ith neuron of the $(l-1)$th layer, and b^l is the bias of the lth layer.

2) FNN MODEL

FNN is a fixed four-layer neural network that combines the advances of fuzzy theory and neural network method. As shown in Fig. 4, FNN consists of input layer, output layer, and two hidden layers called fuzzy layer and rule layer. The output of fuzzy layer is defined as:

$$g_j^i = \exp\left[-\frac{\left(\overline{x_i} - u_{ij}\right)^2}{\sigma^2}\right], \tag{5}$$

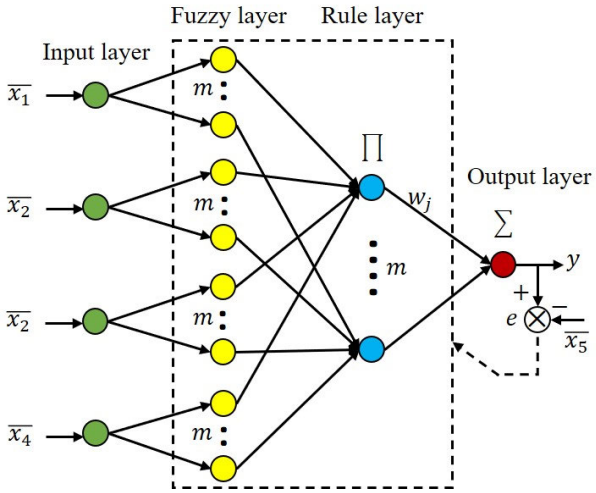

FIGURE 4. General structure of FNN model.

where $\overline{x_i}$ are the input variables, $1 \le i \le n$, u_{ij} is the clustered center of the jth neuron of the ith input, which obtained by k-means method, $1 \le j \le m$, σ is the center width or standard deviation, m and n are the numbers of neurons and inputs respectively. As for FNN, σ and m are structure hyperparameters of the model given by the user.

The output of rule layer is then calculated by the equation as follow:

$$\pi_j = g_j^1 g_j^2 \cdots g_j^n = \prod_{i=1}^{n} g_j^i, \quad 1 \le j \le m. \tag{6}$$

Finally, the output of the model is calculated by

$$y = \sum_{j=1}^{m} w_j \pi_j, \tag{7}$$

where w_j are the weights of the output layer.

3) RBFNN MODEL

The basic form of RBFNN is shown in Fig. 5. The input layer consists of a number of source nodes (sensing units) that connect the network to the external environment. The second layer is the only hidden layer in the network, also known as the radial base layer. It is used for nonlinear transformation from the input space to the hidden space. The output layer is a linear layer.

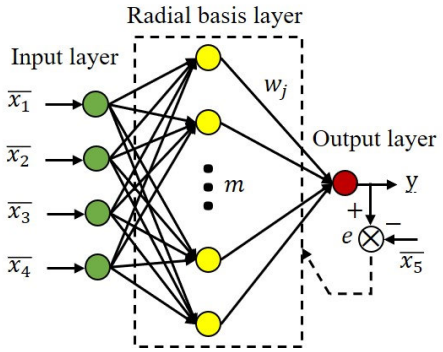

FIGURE 5. General structure of RBFNN model.

In the radial base layer, the Gaussian function is used as the activation function of neurons for nonlinear transformation, which is defined as:

$$g_j\left(\overline{x_i}\right) = exp\left\{\frac{-||\overline{x_i} - \mu_j||^2}{2\sigma^2}\right\}, \tag{8}$$

where $\overline{x_i}$ are the input variables, $i = 1, \ldots, n$, μ_j is the center of the jth node of the radial base layer and is usually calculated by k-means method, $j = 1, \ldots, m$; σ is the width parameter. σ and m are the user-defined parameters. In this article, m is set to the size of the training data by default. Therefore, only σ needs to be determined in RBFNN model.

Finally, the output of the model is calculated by

$$y = \sum_{j=1}^{m} w_j g_j\left(\overline{x_i}\right), \tag{9}$$

where w_j are the weights of the output layer.

B. MODEL TRAINING

The parameters of the ANN surrogate model need to be tuned by gradient-based optimization algorithm to fit the training data accurately, such as the weight w, the bias b, the center μ, and the width σ. This study used adaptive moment (Adam) estimation algorithm [44] to optimize the parameter matrices of the designed models, due to its advantages of high computing efficiency, little memory required, and constant rescaling of the diagonal of the gradient. The code steps of Adam algorithm are summarized as follows:

(1) At the beginning of the algorithm, the following parameters need to be initialized: the objective function $f(\theta)$ (θ is the parameter matrices of the designed models), the initial learning rate η, the initial parameter vector θ_0, and the other initialized parameters, such as exponential decay rates $\beta_1, \beta_2 \in [0, 1)$, timestep $t = 0$, first moment vector $m_0 = 0$, second moment vector $v_0 = 0$, and $\epsilon = 10^{-8}$.

(2) Get gradients with respect to the objective function at timestep t:

$$g_t = \nabla_\theta f_t\left(\theta_{t-1}\right). \tag{10}$$

(3) Update the biased first moment and second raw moment:

$$m_t = \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t, \tag{11}$$
$$v_t = \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2, \tag{12}$$

where g_t^2 indicates the elementwise square $g_t \odot g_t$.

(4) Compute the bias-corrected first moment and second raw moment:

$$\hat{m}_t = m_t/(1 - \beta_1^t), \tag{13}$$
$$\hat{v}_t = v_t/(1 - \beta_2^t). \tag{14}$$

(5) Update the parameters:

$$\theta_t = \theta_{t-1} - \eta \cdot \hat{m}_t/(\sqrt{\hat{v}_t + \epsilon}). \tag{15}$$

(6) If the convergence conditions are satisfied, the resulting parameters θ_t are returned; otherwise, go to step (2).

The process of tuning parameters by gradient-based algorithm is also called error back propagation, that is, the dotted path of e in Figs. 3, 4 and 5. In order to fine-train the model, the training data is usually divided into small batches. The batch size is a user-defined parameter. In this paper, the parameters of the Adam algorithm, as well as the batch size, are collectively referred to as the learning algorithm hyperparameters.

IV. ADAPTIVE MODELING OF ANN

In this section, the proposed MH-TRMPS hyperparameter optimization method and the adaptive ANN surrogate modeling process based on MH-TRMPS will be introduced in detail.

A. MH-TRMPS METHOD

1) MH ALGORITHM

Hyperband [36] is a general-purpose algorithm to draw a random sample from large-scale validation sets. This algorithm only requires knowledge of R and η. R represents the maximum amount of resources that can be allocated to any given hyperparameter configuration, and η is the down-sampling rate depending on R to yield ≈ 5 brackets with a minimum of 3 brackets by $s_{max} = \log_\eta R$. The so-called bracket s ($0 \le s \le s_{max}$) is also the number of global sampling. Concretely, for any s, $n = (s_{max} + 1) \times \frac{\eta^s}{(s+1)}$ points are sampled randomly, and each of points is allocated $r = R\eta^{-s}$ resources to evaluate its function value. Then, a down-sampling process is performed. For each down-sampling, the best $1/\eta$ part will be retained from the n points after the running round.

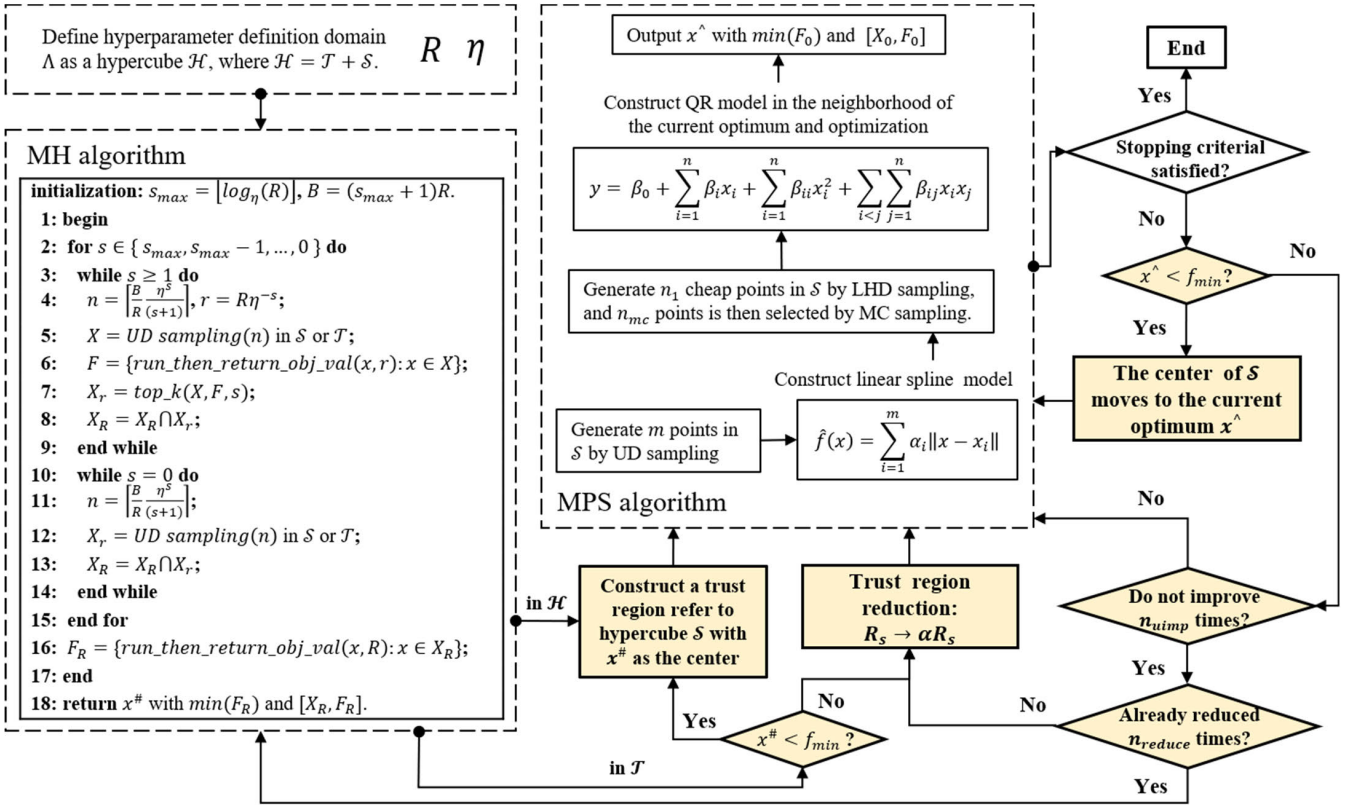

FIGURE 6. The process of MH-TRMPS algorithm.

Hyperband was designed with multiple down-sampling in each bracket to gradually eliminate the early poor performance configurations. However, configurations with early poor performance may not necessarily result in final poor performance. In this situation, multiple down-sampling is futile, as many promising configurations may be randomly dropped at the first down-sampling. Moreover, multiple down-sampling means that black box function needs to be called frequently, which is time-consuming. Consequently, a modified hyperband is proposed.

Similar to hyperband, MH cannot guarantee to find the global optimum. In this situation, more 'intelligence' needs to be built in. A natural method is to construct a small region which centered on the current optimum searched by MH, and then to search the region and find the global optimum by using a model-based method, e.g., MPS. In previous studies, such a small region is often referred to as the trust region [40], [41]. Since a certain number of initial starting points are required in MPS to construct a response surface, and the evaluation of these points is expensive. Therefore, in MH, each bracket is designed to have only one down-sampling. And after down-sampling, the s best configurations in each bracket will be retained, instead of one.

As shown in Fig. 6, the maximal bracket s_{max} and resource burden B are initialized at the beginning of the MH algorithm. Then, uniform design (UD) [45], [46] is executed in the hyperparameter definition domain to obtain n configurations. Note that this is a global random sampling process. UD is used because it has better uniformity than other sampling methods. After obtaining early performance by the function *run_then_return_obj_val* (x, r), the best s configurations are selected by using $top\_k(X, L, \lfloor \frac{n_i}{\eta} \rfloor)$ and retain in X_R. At last, the final performance of each configuration x in X_R is evaluated in *run_then_return_obj_val* (x, R).

2) MH-TRMPS

Before the start of MH-TRMPS, some parameters need to be defined. Let hypercube $\mathcal{H}$ corresponds to the hyperparameter definition space Λ, and hypercube $\mathcal{S}$ corresponds to the trust region which is a subregion of $\mathcal{H}$. The region between $\mathcal{S}$ and $\mathcal{H}$ corresponds to hypercube $\mathcal{T}$, where $\mathcal{T} = \mathcal{H} - \mathcal{S}$. The initial size of $\mathcal{S}$ is set to $R_{\text{s,inintial}} = 0.5$, and the size of $\mathcal{H}$ is set to $R_{\mathcal{H}} = 1$. To avoid overexploitation, the minimal size of $\mathcal{S}$ is set to $R_{\text{s,min}} = 0.01$, and the reduction factor is defined as $\alpha \in [0.5, 0.9]$ (default$\alpha = 0.7$). The details of the MH-TRMPS algorithm are as follows:

Step 1. Initial trust region construction

As shown in Fig. 6, generate $x^\#$ and $[X_R, F_R]$ by MH in the definition space $\mathcal{H}$ and construct the initial trust region $\mathcal{S}$ with current optimum $x^\#$ as the center. Append $[X_R, F_R]$ to $[X, F]$.

Step 2. MPS optimization

Step 2.1. UD sampling. If $x \in X$ within $\mathcal{S}$, place x in X_0. Suppose that n_0 is the number of points in X_0, if $n_0 < 10$, $10 - n_0$ expensive points are sampled from $\mathcal{S}$ by UD, evaluated, and appended to $[X_0, F_0]$.

Step 2.2. Global model construction. The linear spline function is used to fit $[X_0, F_0]$ to obtained a global model which is defined as:

$$\hat{f}(x) = \sum_{i=1}^{m} \alpha_i \|x - x_i\|, \tag{16}$$

where $\hat{f}(x_i) = f(x_i)$, $i = 1, 2, \ldots, m$, $\|\cdot\|$ is the Euclidean norm, m is the number of expensive points x_i, and α_i are the weights.

Step 2.3. Mode pursue sampling. Define $g(\mathrm{x}) = c_0 - \hat{f}(x)$ as the mode function, where c_0 is any constant that ensures $g(\mathrm{x}) \geq 0$. Based on the mode function, the random-discretization-based Monte Carlo sampling method [47] is performed to systematically generate more sample points in the neighborhood of the function mode while statistically covering the entire search space. In one iteration, $n_{mc} = \lceil n_v/2 \rceil$ expensive points will be selected from n_1 (usually a large number) cheap points and added to $[X_0, F_0]$, where n_v is the number of variables. After each iteration, a speed control factor r is used to control the sampling process to balance exploration and exploitation.

Step 2.4. Local model construction. Construct the quadratic regression (QR) model using the $[X_q, F_q]$ from $[X_0, F_0]$, where $[X_q, F_q]$ is the neighborhood closest to the current optimum. The mathematical expression of an n-dimensional generic quadratic model is defined as:

$$y = \beta_0 + \sum_{i=1}^{n} \beta_i x_i + \sum_{i=1}^{n} \beta_{ii} x_i^2 + \sum_{i<j} \sum_{j=1}^{n} \beta_{ij} x_i x_j, \tag{17}$$

where β_i, β_{ii}, and β_{ij} are regression coefficients, x_i represents the expensive points in X_q.

To assess the performance of model fitting, a coefficient of determination R^2 is used. R^2 is defined as:

$$R^2 = \frac{\sum_{i=1}^{n} (\hat{y}_i - \bar{y})^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2}, \tag{18}$$

where $\hat{y}_i$ represents the QR function values, y_i represents the back-box function values, and $\bar{y}$ is the mean of y_i.

Step 2.5. Locally sampling and local model reconstruction. If $R^2 \approx 1$, $n_{nb} = \lceil n_v/3 \rceil$ expensive points are sampled randomly in the neighborhood and evaluated, and added to $[X_q, F_q]$ and $[X_0, F_0]$. Reconstruct the QR model using $[X_q, F_q]$ and recalculate R^2. Calculate *Diff* by the follow equation

$$Diff = max \left\{ \left| f_{fit}^{(i)} - f^{(i)} \right| \right\}, \tag{19}$$

where $f_{fit}^{(i)}$ represents the QR function values, and $f^{(i)}$ represents the actual function values. $i = 1, \ldots, j$ and $j = \frac{(n_v+1)(n_v+2)}{2} + 1 + \frac{n_v}{2}$.

Step 2.6. Optimization based on the local model. If $R^2 \approx 1$ and $Diff < \varepsilon_d$, perform local optimization based on the QR model to generate a global optimal candidate x_0, and evaluate its black box function value. Add (x_0, f_0) to $[X_0, F_0]$.

Step 2.7. Return hyperparameter configuration $x^{\wedge}$ with the minimum function value, $[X_0, F_0]$.

Step 3. Adaptive trust region strategy

Step 3.1. Let $f_{min} = min(F)$. If $x \in X_0$ and $x \notin X$, place (x, f) in $[X, F]$. If $x^{\wedge} < f_{min}$, the center of $\mathcal{S}$ moves to the current optimum, and go to Step 2.

Step 3.2. If the MPS search does not improve for $n_{\text{uimp}} = 2$ consecutive iterations, reduce R_s to αR_s, and go to Step 2.

Step 3.3. If the radius of $\mathcal{S}$ is continuously reduced by $n_{\text{reduce}} = 2$ times, and the MPS search no improvement, search the T using MH to generate $x^{\#}$ and $[X_R, F_R]$. If $x \in X_R$ and $x \notin X$, place (x, f) in $[X, F]$. If $x^{\#} < f_{min}$, $f_{min} = x^{\#}$. A new trust region $\mathcal{S}$ is then reconstructed with $x^{\#}$ as the center, and go to Step 2. Else, reduce R_s to αR_s, and go to Step 2.

Step 3.4. If R_s reaches the preset minimum radius $R_{s,\min}$ or the maximal number of function evaluations is met, stop iteration and return to the current optimum x^* with the smallest intermediate loss seen so far. Else, go to step 2.

Simply put, if the MPS search is consistently reliable, move the location of TR_i so that its center always falls on the current optimum, and try again; if the MPS search failed continuously, reduce the size of TR_i and try again; if the candidate solution does not produce a sufficient increase in the modeling accuracy after multiple reduction of the size of TR_i, $j = j+1$. Meanwhile, MH_j search is conducted in the untrusted region; if MH_j search is reliable, $i = i+1$; where i is the number of times the initial trust region is constructed, and j is the number of times MH is implemented. In the theory, the global optimum can be obtained through the iteration process of MH and MPS, because MH has global sampling characteristics while MPS has global convergence [37].

B. ADAPTIVE MODELING USING MH-TRMPS

Fig. 7 shows the simplified diagram of ANN surrogate adaptive modeling based on MH-TRMPS. There are three parts: input data, hyperparameter selection and model training, and output of the optimal model. To put it another way, the adaptive modeling is essentially a nested optimization process that involves: (1) for each given hyperparameter configuration, using Gradient descent algorithm (e.g., Adam) to optimize the parameters of the ANN to obtained a well-trained surrogate model (inner optimization), (2) using hyperparameter optimization algorithm (e.g., MH-TRMPS) to optimize the hyperparameters to generate a new configuration (outer optimization), and (3) iterating steps (1) and (2). The hyperparameter optimization mathematical expression is defined as:

$$\mathcal{A}_{\lambda^*} \approx \underset{\lambda \in \Lambda}{\text{agrmin}} \mathcal{L}\left(X^{\text{valid}}; \mathcal{A}_\lambda\left(X^{\text{train}}\right)\right). \tag{20}$$

where $\mathcal{A}_\lambda$ is ANN surrogate with the given hyperparameter configuration λ, Λ is hyperparameter definition domain, $\mathcal{L}$ is

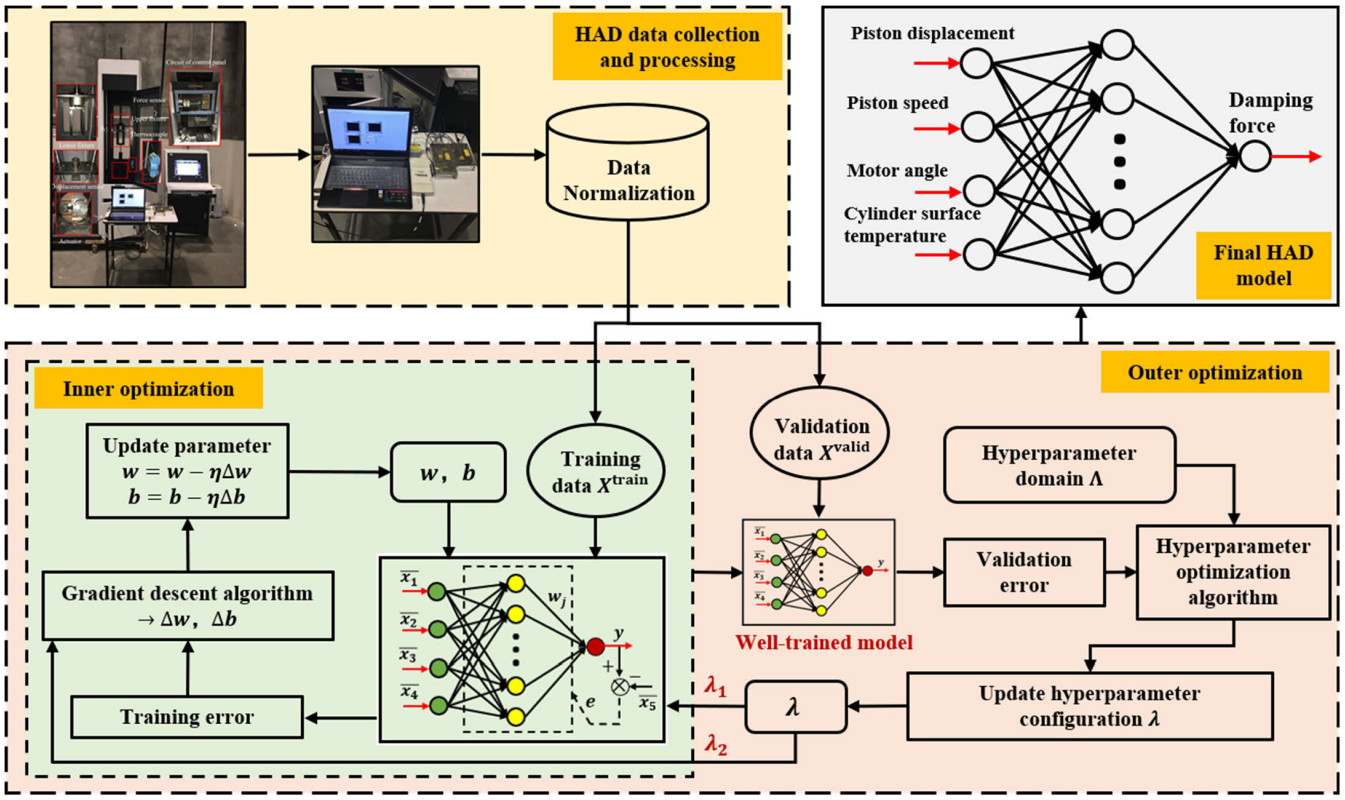

FIGURE 7. The flow diagram of adaptive modeling.

validation error, X^{train} is training data, and X^{valid} is validation data.

Obviously, the evaluation of hyperparameter configuration is time-consuming, because a well-trained model requires multiple iterations to converge. It can also be seen from Fig. 7 that in this work, the structural hyperparameters (λ_1) of the ANN surrogate model need to be determined first, and then the hyperparameters (λ_2) of the learning algorithm are selected to train the model. Moreover, the difficulties of selecting the optimal configuration may increase exponentially as the number of hyperparameters increases. Take MLP as an example. During the cross-validation process, if the termination condition is satisfied, multiple "$\lambda \rightarrow \mathcal{L}$" are obtained. And then, the current optimal hyperparameter configuration λ^* with network parameters w and b will be output as the best model $\mathcal{A}_{\lambda^*}$.

V. NUMERICAL EXPERIMENTS

In this section, all the compared methods will be used to model the unknown regulating mechanism of the nonlinear HAD for the SAS system, and the modeling performance will be given and discussed.

A. EXPERIMENT SETTINGS

The historical collected data from data collection system is designed into three data sets. Each data set is split into a training and validation set: (1) case-1 has 1400 and 87 groups, (2) case-2 has 5000 and 200 groups, and (3) case-3 has 20000 and 3000 groups. Concretely, for case-1, expert method (EXP) [6] is used as the benchmark; for case-2, grid search (GS) [12] is used for comparison; for case-3, RS [28], GP-EI [30], and hyperband [36] are used as the benchmarks. The RS and hyperband are direct stochastic optimization methods, while GP-EI is one of the well-known model-based Bayesian optimization methods.

The down-sampling rate η is set to 3 for hyperband and MH, and the initial start points for model-based methods is set to 10. In this work, $50R$ are used as the total iteration resources to constrain the comparison method, where R is set to 250. The root mean squared error (RMSE) is used to assess the generalization capabilities of the ANN surrogate model and is defined as:

$$RMSE = \sqrt{\frac{1}{n}\sum_{k=1}^{n} E^2} = \sqrt{\frac{1}{n}\sum_{k=1}^{n}\left(y_p^{(k)} - \overline{y_a}^{(k)}\right)^2}, \quad (21)$$

where $y_p^{(k)}$ are the actual outputs and $\overline{y_a}^{(k)}$ are the normalized expected outputs at the kth test point, and n is the number of test points.

All experiments are repeated 10 times. Concretely, a search space is defined for each hyperparameter, from which the configuration is generated one by one to construct ANN surrogate model for cross-validation. Under the constraints of total iteration resources, only 50 hyperparameter configurations are verified in one experiment, and the best result will

TABLE 1. Modeling settings using EXP and GS methods.

Benchmark	Algorithm settings
EXP	The search space of learning rate is set to [1e-6, 1e-3] (default by 0.00015 in Ref. [6]); of the number of neurons in hidden layer is set to [5,100] (default by 12 in Ref. [6]). The batch size is set to 50, and σ is set to 1.
GS	The search space of learning rate is set to [1e-5, 1e-1]; of batch size is set to 2^n, where $n \in [4, 8]$; of the number of hidden neurons is set to [5,500], where the number of hidden layer is 3. Each search space is divided into 5 equal parts for verification.

TABLE 2. The settings for adaptive modeling.

Model	Hyperparameter	Scale	Min	Max
	Structure hyperparameters			
MLP	No. of hidden layers (n)	linear	1	5
	No. of neurons in each hidden layer (k_n)	power	2^4	2^{10}
FNN	σ	linear	0.001	1
	m	power	2^4	2^{10}
RBFNN	σ	linear	0.001	1
	Learning algorithm hyperparameters			
	Learning rate	linear	1e-6	1e-1
	Batch size	power	2^4	2^8
	β_1	linear	0.5	1
	β_2	linear	0.5	1

be retained. When all 10 experiments have been performed, the average value will be calculated and compared.

To ensure that the experiment settings are consistent with those of the original literatures: (1) For EXP, the learning rate and the number of hidden layer neurons are set as adjustable hyperparameters; (2) For GS, learning rate, batch size and number of hidden layer neurons are set as adjustable hyperparameters. More detailed experiment settings are shown in Table 1.

The setting of MH-TRMPS is listed in Table 2 (two columns on the right of Table 2) because it is an adaptive method. Note that the learning algorithm hyperparameters are the same because Adam is used to train all three models. But for different models, the structure hyperparameters are usually different, as shown in Table 2. Generally, the definition of hyperparameter search space affects the optimization efficiency, so it still relies on expert experience to avoid too large a definition domain.

Codes for all compared algorithms can be obtained from shared resources: MPS from the Product Design and Optimization Laboratory (PDOL)[1]; GP-EI[2] and hyperband[3] from an open source project hosting platform GitHub. For random search and grid search, they are very easy to achieve. To ensure the independence of MPS and hyperband, MH-TRMPS is implemented by MATLAB and Python language together, but the compiler environment is python.

B. EXPERIMENT RESULTS

1) THE MODELING RESULTS OF CASE-1

The average damping forces predicted by MLP and FNN surrogate models of the nonlinear HAD are plotted in Figs. 8 and 9. The results of EXP and MH-TRMPS methods are shown in green and red, respectively. Blue represents the expected value. The results show that the model constructed by MH-TRMPS has higher accuracy than the model constructed by EXP method. The reason may be that in the

[1] http://www.sfu.ca/~gwa5/software.html
[2] https://github.com/Mikhail-Naumov/Bayesian_Opt_Testing
[3] https://github.com/zygmuntz/hyperband

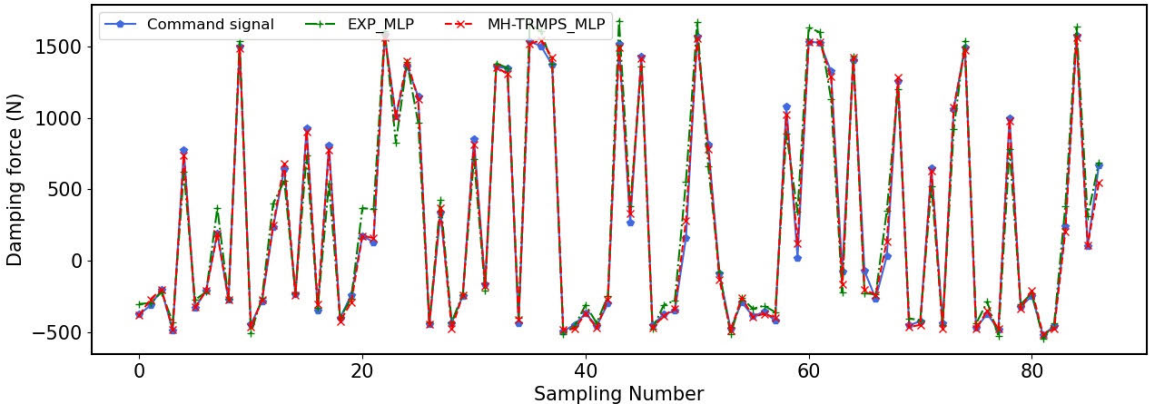

FIGURE 8. Average prediction results from MLP model using EXP and MH-TRMPS.

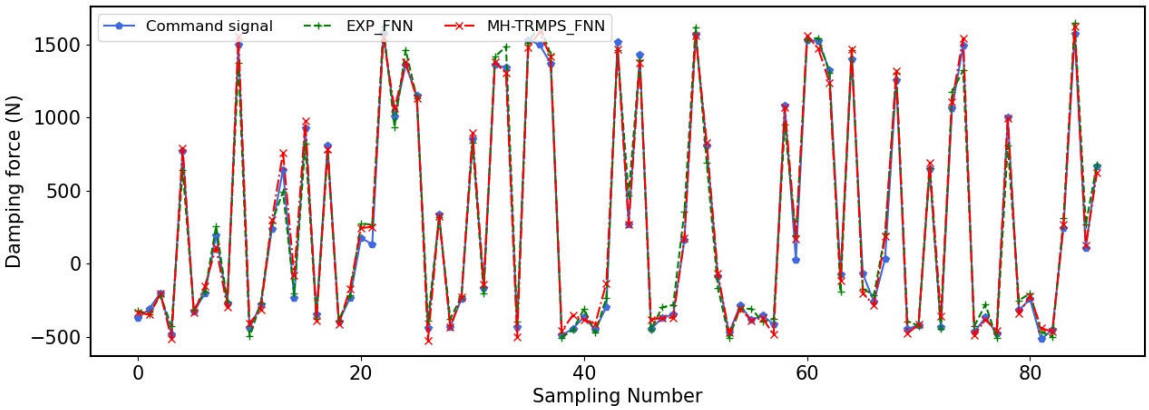

FIGURE 9. Average prediction results from FNN model using EXP and MH-TRMPS.

TABLE 3. Comparison results of EXP and MH-TRMPS on case-1.

Model	EXP (N)	MH-TRMPS（N）	RMSE reduction
MLP	81	**42.1**	48%
FNN	57.4	**36.4**	36.6%

TABLE 4. Comparison results of GS and MH-TRMPS on case-2.

Model	GS (N)	MH-TRMPS (N)	RMSE reduced
MLP	43.5	**37.2**	**14.5%**

adaptive modeling, more hyperparameters are optimized so that the model can accurately approximate the training data.

Apart from the prediction results, the comparison results of modeling using EXP and MH-TRMPS are listed in Table 3. Moreover, the errors between the command signal and the average predicted value are also plotted in Fig. 10. With the offset between positive error and negative error considered, the average value of error is calculated by using absolute value of the predicted results.

Compared with the EXP method, MH-TRMPS algorithm is much better in terms of implementation accuracy of the desired damping force, and the average RMSE is reduced by at least 36.6%. Moreover, the average RMSE of MLP model using MH-TRMPS method is about 42N, which is even less than that of FNN model using EXP method. As shown in Fig. 10, the average prediction error of the model using MH-TRMPS is closer to the 0 axis than that of EXP method. Therefore, it is confident of integrating the adaptive ANN surrogate model with any SAS control algorithms to optimize the performance of the vehicle.

2) THE MODELING RESULTS OF CASE-2

Fig. 11 shows the average prediction damping force by MLP model of the nonlinear HAD. The comparison results are presented in Table 4 and shown in Fig. 12 to respectively describe average RMSE and the error between the command signal and the average predicted value. It can be seen from the results that compared with the GS algorithm, the accuracy of HAD modeling using the MH-TRMPS algorithm is higher, and the RMSE is reduced by 14.5%. The main reason is that the GS needs to divide the search space of

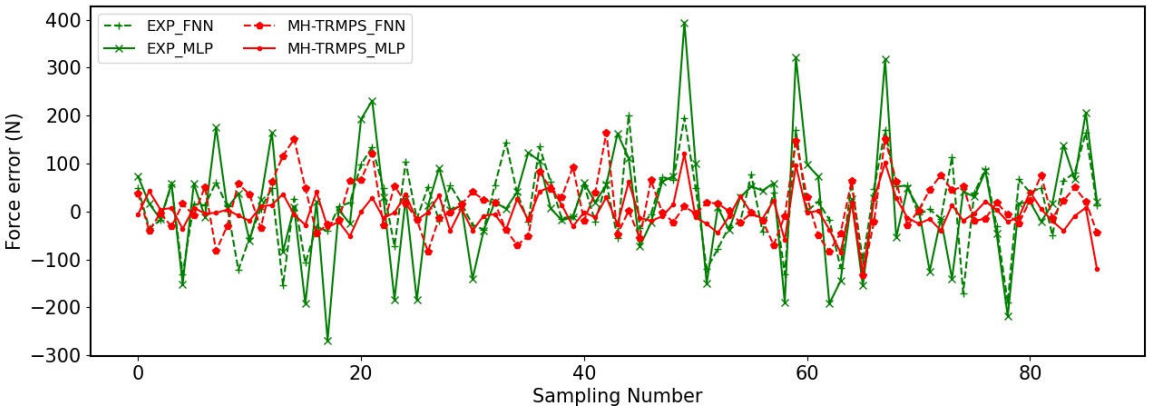

FIGURE 10. Error between average prediction result and command signal on case-1.

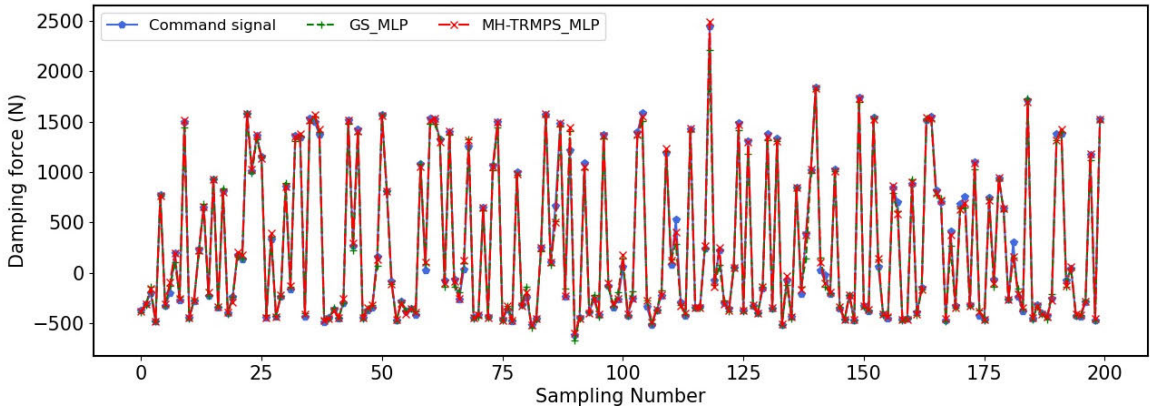

FIGURE 11. Average prediction results from MLP model using GS and MH-TRMPS.

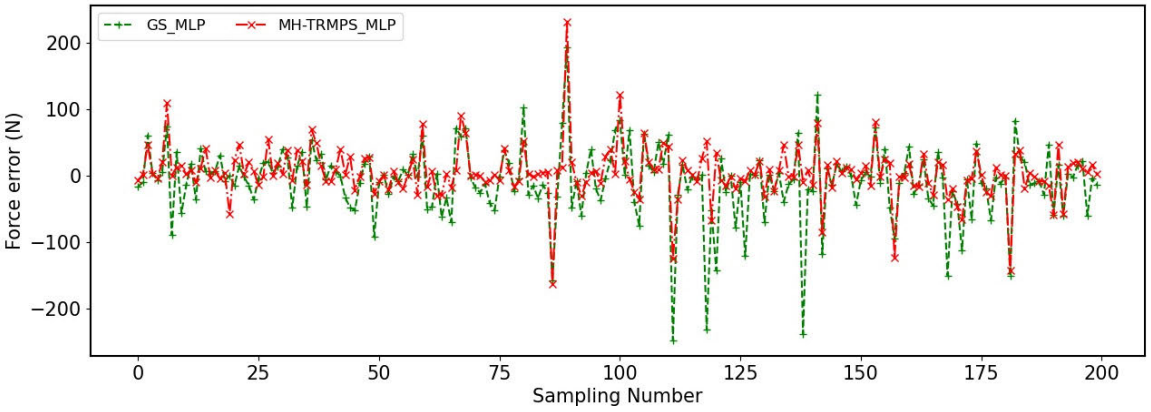

FIGURE 12. Error between average prediction result and command signal on case-2.

each hyperparameter, and then evaluate the combination of the divisions. With limited resources, fine division and exhaustive evaluation are difficult to achieve, especially for high-dimensional problems. In this case, five variables with five divisions are used for testing, which lead to a coarse assessment.

3) THE MODELING RESULTS OF CASE-3

The mean best value of prediction metrics of adaptive modeling found so far as the multiple of R varies are plotted in Fig. 13. It shows that compared with the benchmarks, MH-TRMPS can obtain better modeling performance. For example, the average RMSE of RBFNN model is under $8N$,

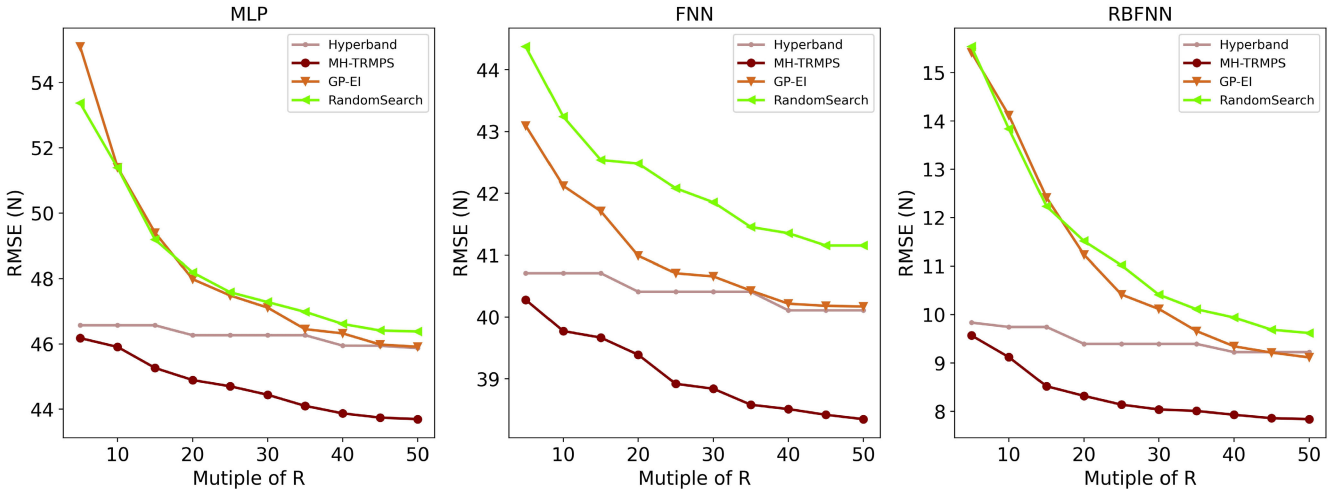

FIGURE 13. Average accuracy of different methods for adaptive modeling.

TABLE 5. The average best results of adaptive modeling using different algorithms.

Model	RS (N)	GP-EI (N)	Hyperband (N)	MH-TRMPS (N)	RMSE reduction	R reduction
MLP	46.38	45.93	*45.89* (12 R)	**43.34**	5.6%	76%
FNN	41.18	40.21	*40.13* (8 R)	**38.11**	5%	84%
RBFNN	9.62	*9.14* (12 R)	9.23	**7.81**	14.6%	76%

Note: Bold item means the best result, italic item means the best result of the benchmark.

while those of the MLP model and the FNN model are about 43N and 38N respectively. This means that in this case, RBFNN is more suitable to model the regulation mechanism of the nonlinear HAD with a high accuracy.

From the multiple of R marked on the horizontal axis of Fig. 13, it also can be seen that after 12R, 8R, and 12R, the mean minimum RMSE of MH-TRMPS is less than those of all the benchmarks that exhausted 50R. This means that the total number of iteration resources for function evaluations of MH-TRMPS is reduced by 76% ($\frac{50-12}{50} \times 100\%$), 84%, and 76% compared to the best benchmark. The best results of adaptive modeling by MH-TRMPS and compared algorithms are listed in Table 5. It can be seen that the average RMSEs of MLP, FNN and RBFNN models constructed by MH-TRMPS are about 43, 38 and 7.8, while the best benchmarks are about 46, 40 and 9.1, which are reduced by about 6% ($\frac{46-43}{46} \times 100\%$), 5%, and 14.6%, respectively.

The experiment also shows that because RBFNN has only one hidden layer, the training speed is very fast. For FNN, its training speed is affected by the number of neurons of the rule layer. Therefore, a trade-off between speed and accuracy is necessary. MLP is a deep neural network. Its training speed is also fast because the activation function is ReLU, which is very simple. However, the modeling accuracy of MLP is not stable enough because a large number of weights and biases need to be initialized randomly.

C. DISCUSSION

As shown in Table 1, the adaptive method can greatly improve the accuracy of the ANN surrogate model of HAD. The main reason may be that in the process of adaptive modeling, more hyperparameters are adaptively adjusted, so that the model can be fine-tuned to accurately approximate the training data. For example, the learning rate is carefully selected in Ref. [6], but the batch size is not given. Furthermore, Table 1 also shows that the prediction error reduction of the MLP model is 48%, which is bigger than that of the FNN model. This is because the number of layers in the FNN model is fixed, while the number of layers in the MLP model can be adaptively selected.

For the case-2 of testing, compared with the semi-adaptive modeling using GS, the adaptive modeling using MH-TRMPS has higher accuracy. The reason may be: In Ref. [12], (1) the number of hidden layers is set to a fixed value, which is 3, and (2) each hyperparameter is searched separately to avoid dimensional disaster. All of these settings are not adaptive enough.

Case 3 is a test of adaptive modeling using different methods and MH-TRMPS obtains the state-of-the-art performance. Results among the methods are slightly different in accuracy, but dramatically different in the iteration resource consuming. For example, under the same validation answer, compared with the best benchmark, the average reduction of

the iteration resources for function evaluation of MH-TRMPS is about 78.7%. But for final accuracy, only an average 8.4% improvement is obtained. This is because in this case, only about six hyperparameters need to be tuned, which is a low-dimensional optimization problem. Therefore, a good set of hyperparameters can be found after each compared algorithm exhausted $50R$ resources.

VI. CONCLUSION

This work presents an adaptive ANN surrogate modeling method based on MH-TRMPS for the HAD problem. Three ANN algorithms are used for testing on three data sets. Among them, the RBFNN is used for the first time to model the nonlinear HAD of SAS system. According to the experimental results, the following conclusions can be drawn:

(1) RBFNN algorithm can be used to model the regulating mechanism of HAD, which has a much higher accuracy when compared with MLP and FNN.

(2) The accuracy of the adaptive method for modeling the nonlinear HAD is higher than manual and semi-adaptive methods in the previous literatures.

(3) Compared with other adaptive methods, MH-TRMPS not only achieves higher modeling accuracy, but also consumes less iteration resources.

Adaptive ANN surrogate modeling based on MH-TRMPS can be found to have excellent performance through many experiments. However, if the early stopping strategy is ineffective for practical issues, the MH will degenerate into random search, which will reduce the optimization efficiency. Future work will parallelize MH-TRMPS for potential expansion related to distributed computing, because: (1) MH has the potential to be parallelized since the global sampling is independent. The most straightforward parallelization scheme is to distribute individual brackets to different machines, which can be done asynchronously. As machines free up, new brackets can be launched with a different set of samples; (2) The processes of MH and MPS are independent, and are suitable for parallelization as well. Moreover, the adaptive model will be applied to the real vehicles in the future work.

REFERENCES

[1] P. Li, J. Lam, and K. C. Cheung, "H_∞ control of periodic piecewise vibration systems with actuator saturation," *J. Vibrat. Control*, vol. 23, no. 20, pp. 3377–3391, Feb. 2016.

[2] H. Pan, X. Jing, W. Sun, and H. Gao, "A bioinspired dynamics-based adaptive tracking control for nonlinear suspension systems," *IEEE Trans. Control Syst. Technol.*, vol. 26, no. 3, pp. 903–914, May 2018.

[3] Y.-J. Liu, Q. Zeng, S. Tong, C. L. P. Chen, and L. Liu, "Adaptive neural network control for active suspension systems with time-varying vertical displacement and speed constraints," *IEEE Trans. Ind. Electron.*, vol. 66, no. 12, pp. 9458–9466, Dec. 2019.

[4] F. Zhao, S. S. Ge, M. Dong, F. Tu, and Y. Qin, "Adaptive neural network control for active suspension system with actuator saturation," *IET Control Theory Appl.*, vol. 10, no. 14, pp. 1696–1705, Sep. 2016.

[5] N. Bhanot, "Artificial neural networks based modeling and analysis of semi-active damper system," M.S. thesis, Dept. Mach. Eng., Virginia Tech, Blacksburg, VA, USA, 2017.

[6] X. Ma, P. K. Wong, and J. Zhao, "Practical multi-objective control for automotive semi-active suspension system with nonlinear hydraulic adjustable damper," *Mech. Syst. Signal Process.*, vol. 117, pp. 667–688, Feb. 2019.

[7] D. L. Guo, H. Y. Hu, and J. Q. Yi, "Neural network control for a semi-active vehicle suspension with a magnetorheological damper," *J. Vibrat. Control*, vol. 10, no. 3, pp. 461–471, Mar. 2004, doi: 10.1177/1077546304038968.

[8] S. Sun, J. Yang, W. Li, H. Deng, H. Du, and G. Alici, "Development of a novel variable stiffness and damping magnetorheological fluid damper," *Smart Mater. Struct.*, vol. 24, no. 8, Aug. 2015, Art. no. 085021, doi: 10.1088/0964-1726/24/8/085021.

[9] X. Sun, C. Yuan, Y. Cai, S. Wang, and L. Chen, "Model predictive control of an air suspension system with damping multi-mode switching damper based on hybrid model," *Mech. Syst. Signal Process.*, vol. 94, pp. 94–110, Sep. 2017.

[10] P. Li, J. Lam, and K. C. Cheung, "Motion-based active disturbance rejection control for a non-linear full-car suspension system," *Proc. Inst. Mech. Eng., D, J. Automobile Eng.*, vol. 232, no. 5, pp. 616–631, Aug. 2017.

[11] J. Zhao, P. K. Wong, X. Ma, and Z. Xie, "Chassis integrated control for active suspension, active front steering and direct yaw moment systems using hierarchical strategy," *Vehicle Syst. Dyn.*, vol. 55, no. 1, pp. 72–103, Oct. 2016.

[12] C. Duchanoy, M. Moreno-Armendáriz, J. Moreno-Torres, and C. Cruz-Villar, "A deep neural network based model for a kind of magnetorheological dampers," *Sensors*, vol. 19, no. 6, p. 1333, Mar. 2019, doi: 10.3390/s19061333.

[13] Y. Qin, F. Zhao, Z. Wang, L. Gu, and M. Dong, "Comprehensive analysis for influence of controllable damper time delay on semi-active suspension control strategies," *J. Vibrat. Acoust.*, vol. 139, no. 3, Jun. 2017, Art. no. 031006, doi: 10.1115/1.4035700.

[14] X. Xu, W. Wang, N. Zou, L. Chen, and X. Cui, "A comparative study of sensor fault diagnosis methods based on observer for ECAS system," *Mech. Syst. Signal Process.*, vol. 87, pp. 169–183, Mar. 2017.

[15] M. Witters and J. Swevers, "Black-box model identification for a continuously variable, electro-hydraulic semi-active damper," *Mech. Syst. Sig. Process*, vol. 24, no. 1, pp. 4–18, Jan. 2010.

[16] Y. Qin, C. He, X. Shao, H. Du, C. Xiang, and M. Dong, "Vibration mitigation for in-wheel switched reluctance motor driven electric vehicle with dynamic vibration absorbing structures," *J. Sound Vibrat.*, vol. 419, pp. 249–267, Apr. 2018.

[17] J. Zhao, P. K. Wong, Z. Xie, X. Ma, and X. Hua, "Design and control of an automotive variable hydraulic damper using cuckoo search optimized pid method," *Int. J. Automot. Technol.*, vol. 20, no. 1, pp. 51–63, Feb. 2019.

[18] N. Palangsavar and A. R. Mamouri, "Stability investigation of hydraulic interconnected suspension system of a vehicle with a quaternion neural network controller," *Iran. J. Mech. Engi. Trans. ISME*, vol. 20, no. 1, pp. 129–151, Dec. 2019.

[19] X. Ma, P. K. Wong, and J. Zhao, "Adaptive regulating of automotive mono-tube hydraulic adjustable dampers using gray neural network-based compensation system," *Proc. Inst. Mech. Eng., D, J. Automobile Eng.*, vol. 233, no. 10, pp. 2532–2545, Sep. 2019, doi: 10.1177/0954407018800580.

[20] N. Hariharana, V. Senthilb, M. Krishnamoorthic, and S. V. Karthicd, "Application of artificial neural network and response surface methodology for predicting and optimizing dual-fuel CI engine characteristics using hydrogen and bio fuel with water injection," *Fuel*, vol. 270, Mar. 2020, Art. no. 117576, doi: 10.1016/j.fuel.2020.117576.

[21] L. Jiao, P. Zhang, R. Min, Y. Luo, and D. Yang, "Research on PMSM sensorless control based on improved RBF neural network algorithm," in *Proc. 37th Chin. Control Conf. (CCC)*, Jul. 2018, pp. 2928–2933, doi: 10.23919/ChiCC.2018.8483974.

[22] H. K. Ghritlahre and R. K. Prasad, "Exergetic performance prediction of solar air heater using MLP, GRNN and RBF models of artificial neural network technique," *J. Environ. Manage.*, vol. 223, pp. 566–575, Oct. 2018.

[23] S. Heo and J. H. Lee, "Fault detection and classification using artificial neural networks," *IFAC-PapersOnLine*, vol. 51, no. 18, pp. 470–475, 2018, doi: 10.1016/j.ifacol.2018.09.380.

[24] Y. Wu, B. Jiang, and N. Lu, "A descriptor system approach for estimation of incipient faults with application to high-speed railway traction devices," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 49, no. 10, pp. 2108–2118, Oct. 2019, doi: 10.1109/TSMC.2017.2757264.

[25] Y. Wu, B. Jiang, and Y. Wang, "Incipient winding fault detection and diagnosis for squirrel-cage induction motors equipped on CRH trains," *ISA Trans.*, vol. 99, pp. 488–495, Apr. 2020, doi: 10.1016/j.isatra.2019.09.020.

[26] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015, doi: 10.1038/nature14539.

[27] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, Jul. 2006, doi: 10.1162/neco.2006.18.7.1527.

[28] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *J. Mach. Learn. Res.*, vol. 13, pp. 281–305, Feb. 2012.

[29] L. Li and A. Talwalkar, "Random search and reproducibility for neural architecture search," 2019, *arXiv:1902.07638*. [Online]. Available: http://arxiv.org/abs/1902.07638

[30] J. Snoek, H. Larochelle, and R. P. Adams, "Practical Bayesian optimization of machine learning algorithms," in *Proc. NIPS*, vol. 12, vol. 2, Dec. 2012, pp. 2960–2968.

[31] F. Hutter, H. H. Hoos, and K. Leyton-Brown, "Sequential model-based optimization for general algorithm configuration," in *Proc. LION*, Jan. 2011, pp. 507–523, doi: 10.1007/978-3-642-25566-3_40.

[32] J. Bergstra, R. Bardenet, Y. Bengio, and B. Keg, "Algorithms for hyper-parameter optimization," in *Proc. NIPS*, Dec. 2011, pp. 2546–2554.

[33] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas, "Taking the human out of the loop: A review of Bayesian optimization," *Proc. IEEE*, vol. 104, no. 1, pp. 148–175, Jan. 2016.

[34] J. Wu, S. Toscano-Palmerin, P. I. Frazier, and A. Gordon Wilson, "Practical multi-fidelity Bayesian optimization for hyperparameter tuning," 2019, *arXiv:1903.04703*. [Online]. Available: http://arxiv.org/abs/1903.04703

[35] F. Berkenkamp, A. P. Schoellig, and A. Krause, "No-regret Bayesian optimization with unknown hyperparameters," 2019, *arXiv:1901.03357*. [Online]. Available: http://arxiv.org/abs/1901.03357

[36] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar, "Hyperband: A novel bandit-based approach to hyperparameter optimization," *J. Mach. Learn. Res.*, vol. 18, no. 1, pp. 1–52, Apr. 2018.

[37] L. Wang, S. Shan, and G. G. Wang, "Mode-pursuing sampling method for global optimization on expensive black-box functions," *Eng. Optim.*, vol. 36, no. 4, pp. 419–438, Aug. 2004.

[38] R. H. Byrd, J. C. Gilbert, and J. Nocedal, "A trust region method based on interior point techniques for nonlinear programming," *Math. Program.*, vol. 89, no. 1, pp. 149–185, Nov. 2000, doi: 10.1007/PL00011391.

[39] E. D. Nino-Ruiz, "Implicit surrogate models for trust region based methods," *J. Comput. Sci.*, vol. 26, pp. 264–274, May 2018, doi: 10.1016/j.jocs.2018.02.003.

[40] A. R. Conn, N. I. Gould, and P. L. Toint, *Trust Region Methods*. Philadelphia, PA, USA: SIAM, 2000, pp. 67–100.

[41] J. Nocedal and S. Wright, "Trust-region methods," in *Numerical Optimization*, 2nd ed New York, NY, USA: Springer, 2006, pp. 64–93.

[42] X. Xue, K. Cheng, Z. Zhang, J. Lin, D. Wang, Y. Bao, M. Wong, and N. Cheung, "Study of art of automotive active suspensions," Presented at the 4th Int. Con. Power Ele. Sys. Appl., Jun. 2011, doi: 10.1109/PESA.2011.5982958.

[43] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proc. AISTATS*, vol. 14, Jan. 2010, pp. 315–323.

[44] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: http://arxiv.org/abs/1412.6980

[45] K. T. Fang, "The uniform design: Application of number-theoretic methods in experimental design," *Acta Mathematicae Applicatae Sinica*, vol. 3, no. 4, pp. 363–372, Nov. 1980.

[46] M. D. McKay, R. J. Beckman, and W. J. Conover, "A comparison of three methods for selecting values of input variables in the analysis of output from a computer code," *Technometrics*, vol. 21, no. 2, pp. 239–245, 1979, doi: 10.2307/1268522.

[47] C. F. James and L. Wang, "A random-discretization based Monte Carlo sampling method and its applications," *Methodol. Comput. Appl. Probab.*, vol. 4, no. 1, pp. 5–25, Mar. 2012.

JINGLIANG LIN received the B.E. degree in mechanical engineering from the Guangdong University of Technology, Guangzhou, China, in 2008, and the M.E. degree in mechanical and electrical engineering from the Guilin University of Electronic Technology, Guilin, China, in 2011. He is currently pursuing the Ph.D. degree with the College of Mechanical and Electrical Engineering, Guangdong University of Technology.

His research interests include optimized design, machine learning, and artificial intelligence.

HAIYAN LI received the B.E. and M.E. degrees in mechanical engineering and the Ph.D. degree in marine engineering from the Huazhong University of Science and Technology, Wuhan, China, in 1997, 2006, and 2010, respectively.

She is currently an Associate Professor with the Department of Mechanical and Electrical Engineering, Guangdong University of Technology. Her research interests include artificial intelligence, machine vision, visual inspection, mechanical and electrical product design and development, design optimization, and deep learning. She has been presided over one National Natural Science Foundation project, since January 2018, and one Youth Fund Project, from January 2015 to December 2017. She has participated in three projects of the National Natural Science Foundation of China. She has published more than a dozen research articles.

YUNBAO HUANG received the B.E., M.E., and Ph.D. degrees in mechanical engineering from the Huazhong University of Science and Technology, Wuhan, China, in 1998, 2001, and 2004, respectively.

From May 2005 to September 2007, he conducted a Postdoctoral Research in the Department of Mechanical, Materials, and Aeronautical Engineering, Illinois Institute of Technology. He is currently a Professor with the Department of Mechanical and Electrical Engineering, Guangdong University of Technology. His research interests include big data-driven product design methods, complex product multidisciplinary integrated design optimization, artificial intelligence deep learning, and machine vision inspection. He presided over three National Natural Science Fund projects, two National 863 projects, more than ten Provincial Science and Technology projects, and published more than 40 articles.

ZEYING HUANG received the B.E. degree in mechanical design and manufacturing and automation from the Guangdong University of Technology, Guangzhou, China, in 2019, where he is currently pursuing the master's degree with the College of Mechanical and Electrical Engineering.

His research interests include the application of deep learning and transfer learning in the field of mechanical engineering.

ZHIQIAN LUO received the master's degree from the University of Macau, Macau.

He is currently working with the Automotive Engineering Institute, Guangzhou Automobile Group Company Ltd. His research interests are semi-active and active suspension control.