Composite Structures 273 (2021) 114285

Contents lists available at ScienceDirect

Composite Structures

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

Transfer learning based variable-fidelity surrogate model for shell buckling prediction

Kuo Tian [a,b], Zengcong Li [a], Jiaxin Zhang [c], Lei Huang [a], Bo Wang [a,*]

[a] Department of Engineering Mechanics, State Key Laboratory of Structural Analysis for Industrial Equipment, Dalian University of Technology, Dalian 116024, China
[b] DUT Artificial Intelligence Institute, Dalian 116024, China
[c] Oak Ridge National Laboratory, Oak Ridge, TN 37830, USA

ARTICLE INFO

Keywords:
Transfer learning based variable-fidelity surrogate model (TL-VFSM)
Shell buckling
Deep neural network
Bayesian optimization
Variable-stiffness composites

ABSTRACT

This paper provides a novel establishment method of the variable-fidelity surrogate model (VFSM) driven by transfer learning for shell buckling prediction problems such as buckling prediction of variable-stiffness composite shells, aiming at achieving higher prediction accuracy than the traditional VFSM constructed by the bridge function. The transfer learning based variable-fidelity surrogate model (TL-VFSM) includes two steps. In the first step, low-fidelity model and high-fidelity model points are calculated and collected into the source dataset and target dataset for transfer learning, respectively. Based on the source dataset, a Deep Neural Network (DNN) model is initialized. Then, the hyperparameter tuning of the DNN model is carried out based on Bayesian Optimization (BO), which uses the target dataset as the validation set. After the hyperparameter pre-tuning, the pre-trained DNN model is obtained. In the second step, early layers of the pre-trained DNN model are remained, while the last layer is replaced. Then, another hyperparameter tuning is carried out to fine-tune the pre-trained DNN model using BO. After that, the transfer learning is finished and the re-trained DNN model is obtained. Through above two steps, the TL-VFSM is constructed. The effectiveness and efficiency of the proposed TL-VFSM are verified by three test functions and two engineering examples of variable-stiffness composite shells and hierarchical stiffened shells. Example results indicate that, using the same computational cost, the proposed TL-VFSM achieves higher prediction accuracy than traditional VFSMs constructed by the bridge function, and it can be a potential method for time-consuming buckling prediction problems with multi-source data or variable fidelities.

1. Introduction

Aiming at improving the computational efficiency, surrogate modelling methods have been widely used in engineering optimizations. Alternative surrogate modelling methods include Radial Basis Function (RBF), Response Surface Method (RSM), Gaussian Process (GP) and Kriging. Many efforts have been made to verify the effectiveness of surrogate modelling methods. Jouhaud et al. [1] constructed a dynamic Kriging surrogate model for shape optimization of a 2D NACA subsonic airfoil. The Kriging surrogate model was employed to accelerate the multidisciplinary design of optimization of the twin-web disk [2]. Rikards et al. [3] employed a data set obtained through finite element simulation to build the RSM surrogate models, which showed compatibility in the prediction of the ultimate load of composite stiffened shells. Nik et al. [4] applied the polynomial regression surrogate modelling technique into the multi-objective optimization of variable-stiffness panels, which significantly reduced the computational cost. Further, various surrogates models were compared regarding the prediction accuracy in predicting the buckling lod of variable-stiffness panels in Ref. [5]. In the field of reliability analysis, Meng et al. [6–8] combined the active learning and importance learning methods with Kriging model, which significantly improved the computational efficiency of reliability analysis without sacrificing computational accuracy. Wang et al. [9] proposed an efficient data-driven streamline stiffener path optimization framework for sparse stiffener layout design of non-uniform curved grid-stiffened composite shells based on artificial intelligence (AI) algorithms. The decision tree-based machine learning method was employed by Wagner et al. [10] to optimize the laminate stacking of composite cylinders for maximum buckling load and minimum imperfection sensitivity. In the actual engineering optimization such as shell buckling problems, the optimization problems are more complex, which are generally

* Corresponding author.
E-mail address: wangbo@dlut.edu.cn (B. Wang).

https://doi.org/10.1016/j.compstruct.2021.114285
Received 4 August 2020; Revised 31 January 2021; Accepted 11 June 2021
Available online 16 June 2021
0263-8223/© 2021 Elsevier Ltd. All rights reserved.

multi-peak, multivariable and highly non-linear [11]. It would be a great challenge to construct a surrogate model with high prediction accuracy for complex optimization problems. To improve the prediction accuracy of the surrogate model, more sampling points need to be generated. However, the calculation of sampling points based on the high-fidelity model (HFM) is generally time-consuming, which causes a huge computational burden in constructing the surrogate model [12].

The variable-fidelity surrogate model (VFSM) is an effective solution to reduce the computational cost of the surrogate model constructed based on HFM points [13–16]. Generally, VFSM is constructed by combining HFM with low-fidelity model (LFM) via the bridge function [16]. VFSM has been widely used in aerodynamic shape optimization [17], post-buckling optimization [18], uncertainty propagation, inference and optimization [19], etc. The co-RBF method [20] and the co-Kriging method [21] were established, aiming at improving the prediction accuracy of VFSM. Han et al. [22] developed an improved VFSM method using the gradient-enhanced Kriging and generalized hybrid bridge function. Based on the support vector regression-based scaling function, an adaptive global VFSM strategy was established by Zhou et al. [23]. In order to achieve the robust establishment of VFSM, Tian et al. [18] proposed a novel VFSM by the two-step adaptive updating approach, which showed higher robustness than the VFSM without updating. An enlightening survey was given in Ref. [24] to introduce the construction methods, applications and development of VFSM. Some representative applications of VFSM include the hull design of autonomous underwater vehicles [25] and the optimization design of laser beam welding parameters [26], etc. Recently, many research works focus on the combination of machine learning methods with VFSM. Based on the active learning method, Zhou et al. [27] proposed an efficient sequential sampling method, which significantly improved the prediction accuracy of VFSM. Tao and Sun [28] employed a deep belief network as the LFM, and applied deep learning based VFSM for robust aerodynamic design optimization, which improved the optimization efficiency obviously. An efficient multi-fidelity co-kriging statistical learning framework was presented in Ref. [29]. Liu et al. [30] proposed a novel multi-fidelity Gaussian process for modeling with diverse data structures, which could effectively extract the low-fidelity information for facilitating the modeling of the high-fidelity output. It can be found from above research works that the combination of machine learning methods with VFSM would be a potential development tendency for VFSM.

Giselle Fernández-Godino et al. [16] summarized common methods for constructing the bridge function of VFMS, including multiplicative bridge function, additive bridge function and comprehensive bridge function. The role of the bridge function is the information fusion between HFM and LFM. Han et al. [22] pointed out that the bridge function is a key element for constructing a VFSM. Tian et al. [18] found that it would be too challenging to construct a bridge function with high prediction accuracy when the relationship between HFM and LFM is highly non-linear, and thus it would lead to a large prediction error for VFSM. In addition, it would be difficult to select an appropriate constructing method of the bridge function from the four common methods due to the lack of prior knowledge about the relationship between HFM and LFM. Therefore, it would need a lot of trial and error for determining the bridge function, which would bring a large computational cost. We notice that there is a similarity between main ideas of the VFSM and the transfer learning method. Transfer learning is a learning approach in which a model that has been trained for one task is used as a starting point to train a model for another similar task. In the VFSM, the low-fidelity surrogate model (LFSM) is constructed based on responses of LFM points firstly, and then it is used as the starting surrogate model. By means of the HFM information, the LFSM is corrected by the bridge function and thus the VFSM is constructed. It can be found, the pre-trained model for one task in the transfer learning method is similar with the LFSM in the VFSM, and the roles of the HFM information in the transfer learning method and VFSM are similar, which is used to correct the pre-trained model in the transfer learning method or the LFSM in the VFSM. The transfer learning method has been commonly used for object detection, image recognition, speech recognition, etc [31]. The advantages of transfer learning are: (1) It is able to train models using relatively little labeled data by leveraging popular models that have already been trained on large datasets; (2) It can dramatically reduce training time and compute resources. With transfer learning, the model does not need to be trained for as many epochs (a full training cycle on the entire dataset) as a new model would require. Considering the similarity between the VFSM and the transfer learning method, the main objective of this paper is to propose a more powerful construction method of VFSM based on the transfer learning method.

The remainder of this paper is organized as follows. Firstly, the traditional VFSM is introduced briefly in Section 2.1. Then, the detailed steps of the proposed transfer learning based variable-fidelity surrogate model (TL-VFSM) are provided in Section 2.2. In order to make comparisons between the proposed TL-VFSM and the traditional VFSM, three simple examples of test functions and two engineering examples of variable-stiffness composite shells and hierarchical stiffened shells are carried out and discussed. In Section 4, the conclusion of this paper is drawn.

2. Transfer learning based variable-fidelity surrogate model

2.1. Brief introduction of traditional variable-fidelity surrogate model (VFSM)

Firstly, the traditional VFSM is introduced briefly and its basic idea is shown in Fig. 1(a). We use the FE models of cylindrical shells with fine meshes and course meshes to illustrate the HFM and the LFM, respectively, and their buckling modes are shown in Fig. 1(a). It can be observed that the HFM achieves higher prediction accuracy in buckling modes than the LFM. This illustration can make readers understand the difference of LFM and HFM clearer. In the traditional VFSM, the LFSM is constructed based on responses of LFM points firstly, and then it is used as the starting surrogate model. By means of the HFM information, the LFSM is corrected by the bridge function and thus the VFSM is constructed. The common bridge functions include multiplicative bridge function, additive bridge function and comprehensive bridge function. The multiplicative bridge function is introduced firstly. The LFSM y_{LFSM} is constructed based on responses of LFM sampling points x_{LFM} predicted by LFM. Further, LFSM is used to predict responses $y_{LFSM}(x_{HFM})$ on HFM sampling points x_{HFM}. The scaling ratio between predicted responses $y_{HFM}(x_{HFM})$ by HFM and predicted responses $y_{LFSM}(x_{HFM})$ by LFSM on HFM sampling points is regarded as the multiplicative bridge function $\rho(x)$,

$$\rho(x) = \frac{y_{HFM}(x_{HFM})}{y_{LFSM}(x_{HFM})} \tag{1}$$

The multiplicative bridge function is employed as the correction approach for the LFSM, and thus the VFSM can be constructed as

$$y_{VFSM} = \rho(x) \cdot y_{LFSM}(x) \tag{2}$$

The additive bridge function $\delta(x)$ can be expressed as

$$\delta(x) = y_{HFM}(x_{HFM}) - y_{LFSM}(x_{HFM}) \tag{3}$$

In this case, the VFSM can be constructed using the additive bridge function,

$$y_{VFSM} = y_{LFSM}(x) + \delta(x) \tag{4}$$

The comprehensive bridge function combines the multiplicative and additive bridge functions [16]. The VFSM constructed by the comprehensive bridge function can be expressed as

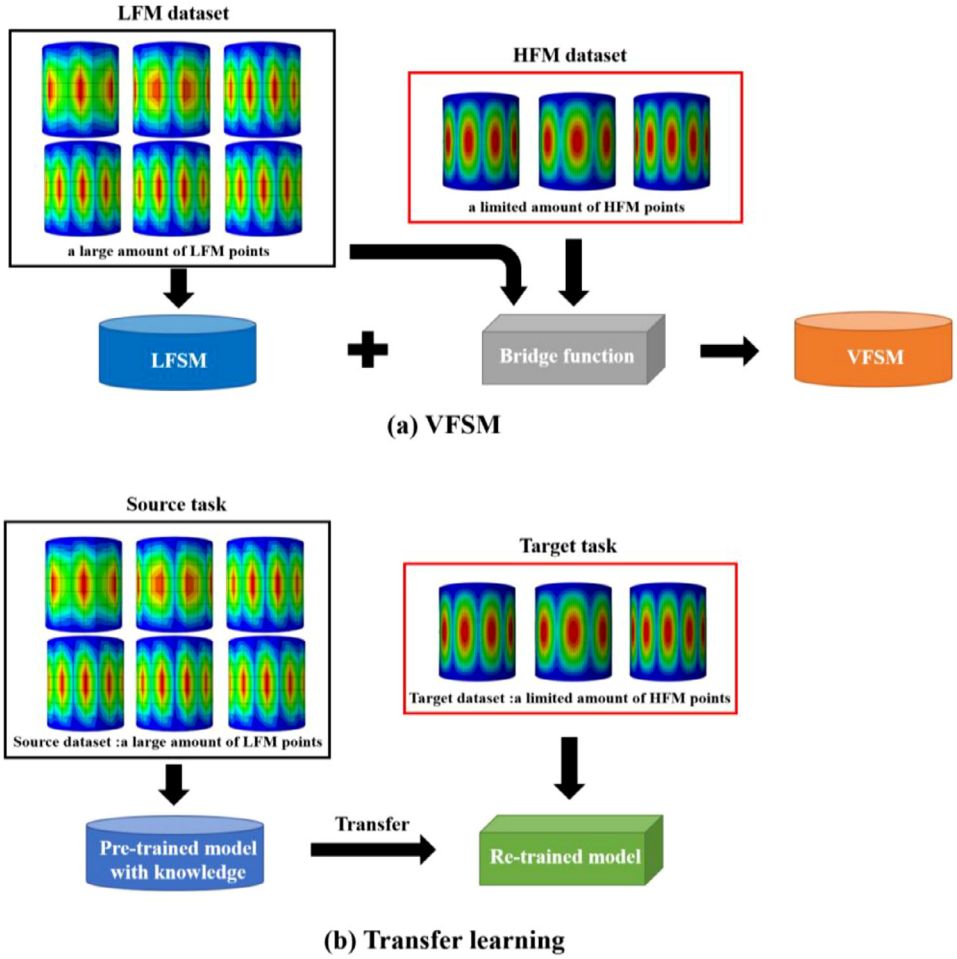

Fig 1. Basic ideas of VFSM and Transfer learning.

$$y_{VFSM} = \omega y_{LFSM}(x) + \delta(x) \tag{5}$$

where ω is the low-fidelity scale factor and can be obtained by

$$\text{find } \omega$$
$$\min \sum_{i=1}^{n_H} [\omega y_{LFSM}(x_i) - y_{HFM}(x_i)]^2 \tag{6}$$

where n_H represents the number of HFM sampling points.

RBF, Kriging or GP algorithms can be employed to construct surrogate models for the LFSM and the bridge function.

2.2. Transfer learning based variable-fidelity surrogate model (TL-VFSM)

Transfer learning is a knowledge learning method that leverages the knowledge learned from a source task to improve learning in a related but different target task. The definition of the transfer learning is as follows.

Definition 1 ((*Transfer learning [31]*).): *Given a source dataset D_s and source task T_s, a target dataset D_t and target task T_t, transfer learning aims to improve the learning of the target predictive function $f_t(\cdot)$ in D_t using the knowledge in D_s and T_s where $D_s \neq D_t$ or $T_s \neq T_t$.*

The basic idea of transfer learning is shown in Fig. 1(b). The knowledge of the source task is extracted from the source dataset composed of a large amount of LFM points, and then a pre-trained model is constructed. After that, the learned knowledge is transferred to a new model. By means of the information of target dataset, the new model is re-trained. Through the comparison of Fig. 1(a) and (b), it can be observed that there are some similarities between the transfer learning method and the VFSM. The pre-trained model for the source task in the transfer learning method is similar with the LFSM in the VFSM, and the roles of the HFM information in the transfer learning method and VFSM are similar, which is used to correct the pre-trained model in the transfer learning method or the LFSM in the VFSM. With respect to the VFSM, it is hard to describe the relationship between HFM and LFM using simple bridge functions such as multiplicative bridge function or additive bridge function when the relationship is highly non-linear. It can be observed from Eqs. (2), (4) and (5) that if the bridge function had a large prediction error, it would lead to a significantly large prediction error for VFSM. In this case, the motivation of this paper is to take full advantage of the strong training ability of the transfer learning method to describe the relationship between HFM and LFM, and thus provide a novel establishment method for VFSM.

In this section, a novel variable-fidelity surrogate model is constructed by transfer learning, referred to as the transfer learning based variable-fidelity surrogate model (TL-VFSM). The schematic diagram of the TL-VFSM is shown in Fig. 2. In Step 1, the pre-trained Deep Neural Network (DNN) model is obtained based on the source dataset composed of a large amount of LFM points. The relationship between the HFM data and the LFM data is non-linear in most cases, which would result in poor performance for the basic linear bridge function. By contrast, the DNN model has strong training and learning abilities, which can capture and predict the non-linear relationship between the HFM data and the LFM data more accurately. In Step 2, the knowledge learned from the pre-trained DNN model is transferred to a new

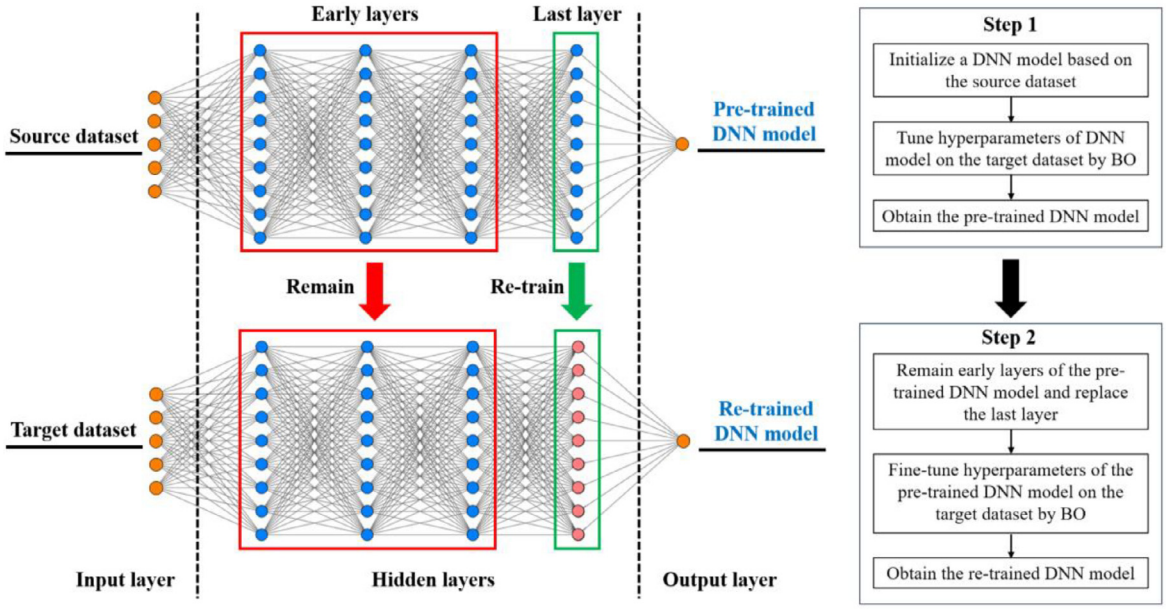

Fig 2. Schematic diagram of transfer learning based variable-fidelity surrogate model (TL-VFSM).

DNN model, and the re-trained DNN model is obtained based on the target dataset composed of a small amount of HFM points. The detailed steps are introduced as follows.

Step 1: Firstly, m LFM points and n HFM points are sampled and calculated, respectively, which are collected into the source dataset and the target dataset for transfer learning. Based on these m LFM points in the source dataset, a DNN model is initialized. The DNN is a neural network composed of many hidden layers. A layer produces the output by using the activation function, and the output from one layer of a DNN model is used as the input to the next layer. The output $z^{(j)}$ from the jth hidden layer of the DNN model can be calculated as

$$z^{(j)} = f(\boldsymbol{W}^{(j)} z^{(j-1)} + \boldsymbol{b}^{(j)}),\ \forall j \in \{1, 2, ..., L\} \tag{7}$$

where $\boldsymbol{W}^{(j)}$, $\boldsymbol{b}^{(j)}$ are the weights and biases which are unknown before the network is trained, d_j is the number of neurons in the jth hidden layer, L is the number of hidden layers and $f(x)$ is the activation function. There are many different forms of the activation function, including the logistic function, the hyperbolic tangent function or the rectified linear unit (ReLU) function [32]. The ReLU function has gained widespread attention in recent years, and can eliminate the need for an unsupervised pretraining phase while training deep architectures [33]. In this paper, the ReLU function is used as the activation function.

To train a DNN model, the purpose is to determine the model parameter values $\boldsymbol{\theta} = \{\boldsymbol{W}^{(j)}, \boldsymbol{b}^{(j)}\}_{j=1}^{L+1}$. $\boldsymbol{\theta}$ contains weights and biases, which can fully describe the structure of the DNN model. The values of $\boldsymbol{\theta}$ can be obtained by minimizing the loss function

$$E(\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N} E_i = \frac{1}{N}\sum_{i=1}^{N} \left(y_i^D - \hat{f}(\boldsymbol{x}_i^D; \boldsymbol{\theta})\right)^2 \tag{8}$$

where $\{\boldsymbol{x}_i^D, y_i^D\}_{i=1}^N$ represents the N sets of input and output of the training data, $\hat{f}(\boldsymbol{x}_i^D; \boldsymbol{\theta})$ represents the output predicted by the DNN model with the model parameter values $\boldsymbol{\theta}$. To minimize the loss function, the Stochastic Gradient Descent (SGD) algorithm [34] is used for optimizing weights and biases of DNN model. Among SGD algorithms, the Adaptive moments (Adam) algorithm [35] has been widely used, which is an improved variant of SGD algorithm. In this paper, the Adam algorithm is employed to train the DNN model. Adam uses a parameter update with an added momentum term, and can keep an element-wise moving average of both the parameter gradients and their squared values,

$$\nabla E(\boldsymbol{\theta}_l) = \frac{\partial E_{i_l}}{\partial \boldsymbol{\theta}_l} \tag{9}$$

$$\boldsymbol{m}_l = \beta_1 \boldsymbol{m}_{l-1} + (1-\beta_1)\nabla E(\boldsymbol{\theta}_l) \tag{10}$$

$$\boldsymbol{v}_l = \beta_2 \boldsymbol{v}_{l-1} + (1-\beta_2)[\nabla E(\boldsymbol{\theta}_l)]^2 \tag{11}$$

where l is the iteration number, $\nabla E(\boldsymbol{\theta}_l)$ is the gradient of the loss function, $\boldsymbol{m}$ is the momentum vector and $\boldsymbol{v}$ is the squared gradient vector. β_1 and β_2 are the decay rate of gradient and decay rate of squared gradient, respectively. Adam uses the moving averages to update the network parameters as

$$\boldsymbol{\theta}_{l+1} = \boldsymbol{\theta}_l - \frac{\alpha \boldsymbol{m}_l}{\sqrt{\boldsymbol{v}_l} + \varepsilon} \tag{12}$$

where ε is a small number to avoid division by zero and α is the learning rate. When the gradients over continuous iterations are similar, a moving average of the gradient is used to enable the parameter updates to pick up momentum in a certain direction. If there is mostly noise in the gradients, then the moving average of the gradient becomes smaller so that the parameter updates become smaller too. After the training by Adam, the weights and biases of the DNN model are obtained.

Algorithm 1: Adam algorithm

input: learning rate α, $0 < \beta_1, \beta_2 < 1$
initialization: $m_0 = 0$, $v_0 = 0$, $\theta_1 = \theta_0$
for $l = 1, 2, \ldots l_{max}$, **do**
    Compute stochastic gradient $\nabla E(\boldsymbol{\theta}_l)$
    $\boldsymbol{m}_l = \beta_1 \boldsymbol{m}_{l-1} + (1-\beta_1)\nabla E(\boldsymbol{\theta}_l)$
    $\boldsymbol{v}_l = \beta_2 \boldsymbol{v}_{l-1} + (1-\beta_2)[\nabla E(\boldsymbol{\theta}_l)]^2$
    $\boldsymbol{\theta}_{l+1} = \boldsymbol{\theta}_l - \frac{\alpha \boldsymbol{m}_l}{\sqrt{\boldsymbol{v}_l} + \varepsilon}$
end

Besides the model parameters like weights and biases discussed above, the prediction accuracy of the DNN model is also significantly affected by hyperparameters. In order to select the optimal model structure for the DNN model automatically, the hyperparameter tuning is carried out using Bayesian Optimization (BO) [36,37]. The optimization variables $\boldsymbol{x}$ of BO compose of multiple hyperparameters (number of neurons in each layer, number of layers, learning rate, etc.), which are set prior to the model parameter training process by Adam algorithm and determine the structure of the DNN model. In BO, m LFM points in the source dataset and n HFM points in the target dataset are used as the training set and validation set, respectively. The optimization objective of BO is to minimize the validation error, which is regarded as the objective function f of BO. Since evaluating the objective function f at $\boldsymbol{x}$ is very expensive, f is approximated using a probabilistic surrogate model that is much cheaper to evaluate. In the BO process, Gaussian process (GP) is used as the probabilistic surrogate model, which can provide a convenient and powerful prior distribution on functions. The GP can be expressed as

$$f(\boldsymbol{x}) \sim \mathrm{GP}(\zeta(\boldsymbol{x}), \gamma(\boldsymbol{x}, \boldsymbol{x}')) \tag{13}$$

where f is the objective function mentioned above which is subjected to Gauss distribution, ζ is the mean function and γ is the covariance function. GP returns the mean and variance of a normal distribution over the possible values of f at $\boldsymbol{x}$.

In BO, the acquisition function μ is used to construct a utility function from the model posterior, which determines the next point to evaluate. Expected Improvement (EI) is employed as the acquisition function in this paper. The BO is an iterative process and its main idea is as follows: (1) The next trial point x_{k+1} is determined according to the current GP model by optimizing the acquisition function μ. (2) With the determined hyperparameters x_{k+1}, the model parameters are trained by Adam algorithm thus the objective function $f(x_{k+1})$ is evaluated. (3) The GP model is updated based on the new data point $(x_{k+1}, f(x_{k+1}))$, and then goes back to step (1). BO samples trial points sequentially, and each trial point is sampled utilizing all the information in the history (reflected by the built surrogate model). Namely, which point will be sampled next is determined by all of those sampled previously. The optimal result can be found by use of BO with N iterations. An illustration of the specific steps of this algorithm is shown in Algorithm 2. After the hyperparameter tuning by BO, the optimal hyperparameters are determined and thus the pre-trained DNN is obtained.

Algorithm 2: Bayesian optimization

1: **initialization:** $D_0 = \phi$
2: **for** $k = 1, 2, \ldots N$, **do**
3: find x_{k+1} by optimizing the acquisition function μ: $x_{k+1} = \arg\max_x \mu(x|D_k)$
4: train the model parameters $\boldsymbol{\theta}$ by Adam algorithm under the hyperparametersx_{k+1}
5: evaluate the objective function: $f(x_{k+1})$
6: $D_{k+1} = D_k \cup \{(x_{k+1}, f(x_{k+1}))\}$
7: update GP model using D_{k+1}
8: **end**

Step 2: Refs. [38,39] demonstrated that the weight parameters of the multi-layer DNN would capture low-level features in the lower hidden layer and high-level features in the higher hidden layer. Jang et al. [38] also pointed out that the estimated DNN weights showed spatial patterns that are remarkably task-specific, particularly in the higher layers. Therefore, the transfer learning method generally remained the early layers of the pre-trained DNN model but re-initialized and re-trained the last layers [38–41]. As for how many last layers should be re-initialized and re-trained, Lu et al. [40] pointed out that for the target task model, only the last classification layer needs to be retrained and Cireşan et al. [41] also stated that for the new task only the last classification layer needs to be re-trained. That is to say, the last layer is suggested to be re-trained. Then, another hyperparameter tuning is carried out by BO to fine-tune the pre-trained DNN model on the target dataset. The hyperparameters involved in the fine-tuning include learning rate, decay rate of gradient, gradient threshold, size of the mini-batch, etc. The learning rate factor of the new last layer is set to be larger than that of the early layers. The target dataset composed of n HFM points generated in Step 1 is reused as the training set for the re-trained DNN model in this step. The k-Fold cross validation is carried out to evaluate the validation error of the re-trained DNN model. n HFM points in the target dataset are divided into k sets. The 1st set of HFM points is used as the validation set, and other (k-1) sets of HFM points involve in training the DNN model. The validation error of the 1st validation set is calculated. This process is repeated k times, and each time the model parameters are determined by Adam algorithm and the mean validation error is calculated, which is regarded as the objective function of BO. After the hyperparameter fine-tuning, the transfer learning is finished and the re-trained DNN model is obtained. Through above two steps, the TL-VFSM is constructed.

In order to verify the prediction accuracy of the constructed TL-VFSM, another set of sampling points is generated as the test set and then calculated by HFM and TL-VFSM, respectively. The prediction error of the test set can be evaluated according to Relative Root Mean Square Error (*RRMSE*) and R^2 values. The formulas are as follows,

$$RRMSE = \frac{\sqrt{\frac{1}{K}\sum_{i=1}^{K}(y_i - \hat{y}_i)^2}}{\sqrt{\frac{1}{K-1}\sum_{i=1}^{K}(y_i - y^{\sim}{}_i)^2}} \tag{14}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{K}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{K}\left(y_i - \tilde{y}_i\right)^2} \tag{15}$$

where, K is the total number of sampling points of the testing set, y_i is the actual value of the ith sampling point by HFM, $\hat{y}_i$ is the predicted value by TL-VFSM and $\tilde{y}_i$ is the mean value of the actual values of all sampling points by HFM. The less *RRMSE* is, the more accurate the TL-VFSM is. The more R^2 approximates to 1.0, the more accurate the TL-VFSM is.

3. Illustrative examples

3.1. Test function 1: Dette & Pepelyshev function (8 dimensions)

In order to verify the effectiveness of the proposed TL-VFSM, an eight-dimensional function (Dette & Pepelyshev function) is firstly studied [42]. The formulas of the HFM and the LFM for the Dette & Pepelyshev function are as follows,

$$f_h(x) = 4(x_1 - 2 + 8x_2 - 8x_2^2)^2 + (3 - 4x_2)^2 + 16 \times \sqrt{x_3 + 1}(2x_3 - 1)^2 + \sum_{i=4}^{8} i\ln(1 + \sum_{j=3}^{i} x_j) \tag{16}$$

$$f_l(x) = f_h(0.85x_1, 0.85x_2, 0.85x_3, 0.85x_4, 1.15x_5, 1.15x_6, 1.15x_7, 1.15x_8) \tag{17}$$

where $x_1, x_2, \ldots, x_8 \in [0, 1]$.

200 LFM and 30 HFM points are generated using the LHS method in the design space and then calculated, respectively. The VFSMs are constructed based on RSM, RBF and Kriging, respectively. Particularly, the additive bridge function, the multiplicative bridge function and the

comprehensive bridge function are used to predict the relationship between HFM and LFM, respectively. Another set of 500 HFM points are generated as the test set for VFSM. The prediction accuracy results of VFSMs are listed in Table 1 and Fig. 3. As for RBF-VFSMs and Kriging-VFSMs, the prediction accuracy results of VFSMs based on different bridge functions are almost the same, which indicates that different bridge functions have small impact on the prediction accuracy of RBF-VFSMs and Kriging-VFSMs. By contrast, the prediction accuracy results of RSM-VFSMs based on different bridge functions are quite different. Particularly, it can be found that the *RRMSE* result (0.513) of RSM-VFSM (comprehensive bridge function) is lower by 14% than that (0.597) of RSM-VFSM (multiplicative bridge function), which demonstrates that the bridge function would influence the prediction accuracy of RSM-VFSMs significantly. Above all, we can infer that due to the lack of prior knowledge about the relationship between HFM and LFM, the bridge function is not accurate or robust in some cases. It can be observed from Table 1 and Fig. 3, R2 values and *RRMSE* values result in similar conclusions about the prediction accuracy, which has been verified in Ref. [18]. For the sake of succinct statement, we take the results of *RRMSE* values into the primary consideration in the following study, and the results of R^2 values are regarded as a reference.

Then, the proposed TL-VFSM is constructed for comparison. In Step 1 of TL-VFSM, a DNN model is constructed based on the source task composed of the 200 LFM points. Adam algorithm is employed to train the weights and biases of the DNN model. The hyperparameters of the DNN model are tuned using BO. In BO, the source task composed of 200 LFM points and the target task composed of 30 HFM points are used as the training set and validation set, respectively. The optimization objective of BO is to minimize the validation error. The maximum iteration number N of BO is 80. The iteration history of BO is displayed in Fig. 4 and it can be observed that the BO converges quickly. The normalization factor for validation error in Fig. 4 is 1072.02. The optimized results of hyperparameters of the pre-tuning are listed in Table 2. The size of the mini-batch is a discrete variable, including 32, 64 and 128. In Step 2 of TL-VFSM, early layers of the pre-trained DNN model are remained, while the last layer of the pre-trained DNN model is replaced. Then, another BO hyperparameter tuning is used to fine-tune the pre-trained DNN model based on the target task. The 5-Fold cross validation is carried out to evaluate the mean validation error of the re-trained DNN model, which is regarded as the objective function of BO. The iteration history of BO is displayed in Fig. 4. The optimized results of hyperparameters of the fine-tuning are listed in Table 3. After the transfer learning, the re-trained DNN model is obtained. The TL-VFSM is constructed by above two steps. Firstly, the influence of the number of last layers to be retrained on the performance of the DNN model is studied. The prediction accuracy results of the DNN models re-training the last one layer, re-training the last two layers and without re-training any last layer are compared, with results shown in Table 1. It can be found that the DNN model re-training the last one layer achieves the highest prediction accuracy. This conclusion agrees well with that in Refs. [40,41]. The prediction error result (*RRMSE* result) of the TL-VFSM is listed in Table 1 and Fig. 3, decreasing by 54%, 38% and 21% than prediction accuracy results of RSM-VFSMs, RBF-VFSMs and Kriging-VFSMs using the same number of HFM and LFM points, respectively.

Table 1
The prediction accuracy of VFSMs (30HFM + 200LFM) for Dette & Pepelyshev function.

	RRMSE	R^2
RSM-VFSM (additive bridge function)	0.527	0.722
RSM-VFSM (multiplicative bridge function)	0.597	0.642
RSM-VFSM (comprehensive bridge function)	0.513	0.736
RBF-VFSM (additive bridge function)	0.392	0.846
RBF-VFSM (multiplicative bridge function)	0.383	0.853
RBF-VFSM (comprehensive bridge function)	0.377	0.857
Kriging-VFSM (additive bridge function)	0.317	0.899
Kriging-VFSM (multiplicative bridge function)	0.300	0.909
Kriging-VFSM (comprehensive bridge function)	0.299	0.910
TL-VFSM (re-train the last one layer using HFM dataset)	**0.235**	**0.945**
TL-VFSM (re-train the last two layers using HFM dataset)	0.347	0.879
TL-VFSM (without re-training any last layer)	0.501	0.748
DNN model (trained by HFM dataset only)	0.665	0.560

Then, the prediction accuracy results of VFSMs using or without BO hyperparameter tuning are compared. It can be found from the results shown in Table 4 that the *RRMSE* result (0.235) of VFSM using BO hyperparameter tuning is smaller by 43.5% than that (0.416) without BO hyperparameter tuning, which indicates that using BO hyperparameter tuning can obtain higher prediction accuracy. In addition, the performance of using the basic DNN model with HFM dataset as the training set is studied, and its prediction error results are listed in Table 1. Through the comparison of the basic DNN model (HFM dataset only) and the proposed TL-VFSM, it can be found that the R^2 result of the basic DNN model (HFM dataset only) is only 0.560, whose prediction accuracy is not good enough. By contrast, the R^2 result of the TL-VFSM is 0.945, which indicates that the TL-VFSM can provide higher prediction accuracy.

Through this example, the high prediction accuracy of the proposed TL-VFSM is verified.

3.2. Test function 2: Convex function (8 dimensions)

Next, an eight-dimensional function (Convex function [43]) is used to investigate the effectiveness of the proposed TL-VFSM in predicting the nonlinear mixed-integer problem. The formulas of the HFM and the LFM for the Convex function are as follows,

$$f_h(x) = 3.1x_1^2 + 7.6x_2^2 + 6.9x_3^2 + 0.004x_4^2 + 19x_5^2 + 3x_6^2 + x_7^2 + 4x_8^2 \quad (18)$$

$$f_l(x) = f_h(x) + x_1x_2^2 - x_3x_8^2 + x_5x_7 - 0.01x_4^2 \quad (19)$$

where integer variables x_1, x_2, x_3, x_4 and continuous variables x_5, x_6, x_7, $x_{8\in}[-10, 10]$.

Since the analysis steps of TL-VFSM for this example is quite similar with those for the example in Section 3.1, this section is introduced briefly to keep the conciseness of this paper. 300 LFM and 30 HFM points are generated using the LHS method in the design space and calculated, respectively, which are then collected into the source task and target task. The maximum iteration number N of BO is 80. The iteration history of BO is displayed in Fig. 5 and the normalization factor for validation error is 8,500,000. The optimized results of hyperparameters of the pre-tuning are listed in Table 5. The iteration history of BO is displayed in Fig. 5. The optimized results of hyperparameters of the fine-tuning are shown in Table 6. Another set of 500 HFM points are generated as the test set for TL-VFSM. The prediction accuracy result of the TL-VFSM is listed in Table 7 and Fig. 6. For comparison, the RSM-VFSM, RBF-VFSM and Kriging-VFSM are constructed based on the same number of HFM and LFM points as TL-VFSM. It should be pointed out that only the additive bridge function is employed in the following study. The prediction accuracy results of RSM-VFSM, RBF-VFSM and Kriging-VFSM are also listed in Table 7 and Fig. 6. By contrast, the prediction accuracy result (*RRMSE* result) of TL-VFSM decreases by 15%, 11% and 5% than prediction error results of RSM-VFSM, RBF-VFSM and Kriging-VFSM, respectively. Thus, the high prediction accuracy of the proposed TL-VFSM has been demonstrated through this mixed-integer example.

3.3. Test function 3: Powell function (16 dimensions)

In order to verify the effectiveness of the proposed TL-VFSM for high-dimensional problems, a 16-dimensional function (Powell func-

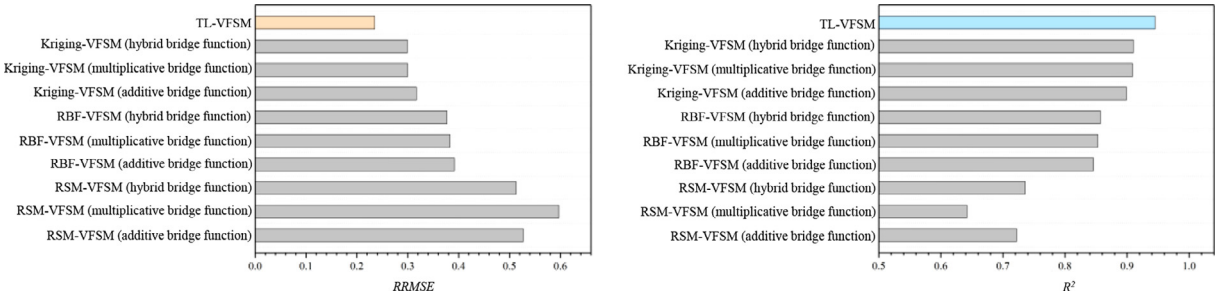

Fig 3. Histograms of prediction accuracy of VFSMs (30HFM + 200LFM) for Dette & Pepelyshev function.

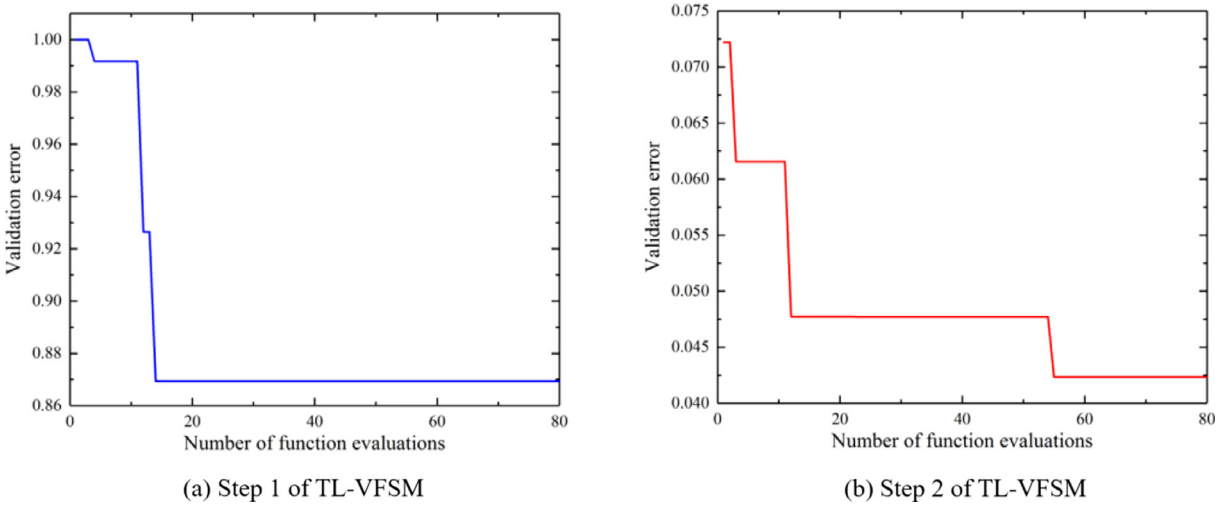

Fig 4. Iteration histories of Bayesian optimizations for TL-VFSM (30HFM + 200LFM) for Dette & Pepelyshev function.

Table 2
Hyperparameter results of pre-tuning for TL-VFSM (30HFM + 200LFM) for Dette & Pepelyshev function.

Hyperparameters	Lower bound	Upper bound	Tuning result (without BO hyperparameter tuning)	Tuning result (using BO hyperparameter tuning)
Number of neurons in each layer	20	200	100	51
Number of layers	2	15	5	3
Learning rate	1e−4	0.1	0.01	0.0144
Decay rate of gradient	0.7	0.98	0.9	0.874
Dropout rate	0.2	0.8	0.5	0.694
Gradient threshold	0.1	5	2.5	3.528
Size of the mini-batch	32	128	64	64
Factor for L_2 regularization	1e−10	1e−2	1e−9	9.661e−10

Table 3
Hyperparameter results of fine-tuning for TL-VFSM (30HFM + 200LFM) for Dette & Pepelyshev function.

Hyperparameters	Lower bound	Upper bound	Tuning result (without BO hyperparameter tuning)	Tuning result (using BO hyperparameter tuning)
Learning rate	1e−8	1e−2	1e−3	9.28e−4
Decay rate of gradient	0.7	0.98	0.9	0.918
Gradient threshold	0.1	5	1	0.455
Size of the mini-batch	32	128	64	64
Factor for L_2 regularization	1e−10	1e−2	1e−8	4.285e−8
Learning rate factor of weight (the last layer)	1	300	100	198
Learning rate factor of bias (the last layer)	1	300	100	197

Table 4
Comparison of the prediction accuracy of VFSMs using or without BO hyperparameter tuning.

	RRMSE	R^2
TL-VFSM (using BO hyperparameter tuning)	0.235	0.945
TL-VFSM (without BO hyperparameter tuning)	0.416	0.830

tion [44]) is tested. The formulas of the HFM and the LFM for the Powell function are as follows,

$$f_h(x) = \sum_{i=1}^{4}[(x_{4i-3} + 10x_{4i-2})^2 + 5(x_{4i-1} - x_{4i})^2 + (x_{4i-2} - 2x_{4i-1})^4 + 10(x_{4i-3} - x_{4i})^4] \quad (20)$$

$$f_l(x) = f_h(0.8x_1, 0.8x_2, 0.8x_3, 0.8x_4, 0.8x_5, 1.2x_6, 1.2x_7, 1.2x_8, 1.2x_9, 1.2x_{10}, 1.2x_{11}, 1.1x_{12}, 1.1x_{13}, 1.1x_{14}, 1.1x_{15}, 1.1x_{16}) \quad (21)$$

where variables x_1, …, $x_{16} \in [-4, 5]$.

Since the dimension of the Powell function is higher than the previous test functions, more LFM and HFM points are needed for constructing the VFSM. 1500 LFM and 150 HFM points are sampled as the source task and target task for VFSM. The maximum iteration number N of BO is 100. The iteration history of BO is displayed in Fig. 7 and the normalization factor for validation error is 3.49e + 7. The optimized results of hyperparameters of the pre-tuning are listed in Table 8. The iteration history of BO is displayed in Fig. 7. The optimized results of hyperparameters of the fine-tuning are shown in Table 9. Another set of 1500 HFM points are generated as the test set for TL-VFSM. The prediction accuracy results of the TL-VFSM are listed in Table 10 and Fig. 8. The prediction accuracy results of the RSM-VFSM, RBF-VFSM and Kriging-VFSM with the same number of HFM and LFM points are listed in Table 10 and Fig. 8. By contrast, the prediction accuracy result (*RRMSE* result) of TL-VFSM decreases by 15%, 10% and 76% than prediction error results of RSM-VFSM, RBF-VFSM and Kriging-VFSM, respectively. Particularly, the R^2 result of RSM-VFSM is negative, which means that the prediction accuracy of RSM-VFSM is too low. Through above comparison, the effectiveness of the proposed TL-VFSM in predicting high-dimensional problems has been verified.

3.4. Engineering example 1: variable-stiffness composite shells (7 design variables)

An engineering example of the variable-stiffness composite shell with 7 design variables is presented. In recent years, variable-stiffness composite shells have been widely used owing to the rapid development of Automatic Fiber Placement (AFP) technology [45]. Many efforts have been made for the analysis and optimization of variable-stiffness composite plates and shells [46–54], which achieved higher mechanical performance (e.g. buckling load, fundamental frequency) than constant-stiffness composite plates and shells. As pointed

Table 5
Hyperparameter results of pre-tuning for TL-VFSM (30HFM + 300LFM) for Convex function.

Hyperparameters	Lower bound	Tuning result	Upper bound
Number of neurons	20	149	200
Number of layers	2	6	15
Learning rate	1e−4	0.007	0.1
Decay rate of gradient	0.7	0.854	0.98
Dropout rate	0.2	0.783	0.8
Gradient threshold	0.1	1.580	5
Size of the mini-batch	32	32	128
Factor for L_2 regularization	1e−10	1.775e−8	1e−2

Table 6
Hyperparameter results of fine-tuning for TL-VFSM (30HFM + 300LFM) for Convex function.

Hyperparameters	Lower bound	Tuning result	Upper bound
Learning rate	1e−8	2.476e−4	1e−2
Decay rate of gradient	0.7	0.970	0.98
Gradient threshold	0.1	0.702	5
Size of the mini-batch	32	64	128
Factor for L_2 regularization	1e−10	1.22e−7	1e−2
Learning rate factor of weight (the last layer)	1	159	300
Learning rate factor of bias (the last layer)	1	37	300

Table 7
The prediction accuracy of VFSMs (30HFM + 300LFM) for Convex function.

	RRMSE	R^2
RSM-VFSM	0.426	0.818
RBF-VFSM	0.409	0.833
Kriging-VFSM	0.414	0.829
TL-VFSM	**0.364**	**0.867**

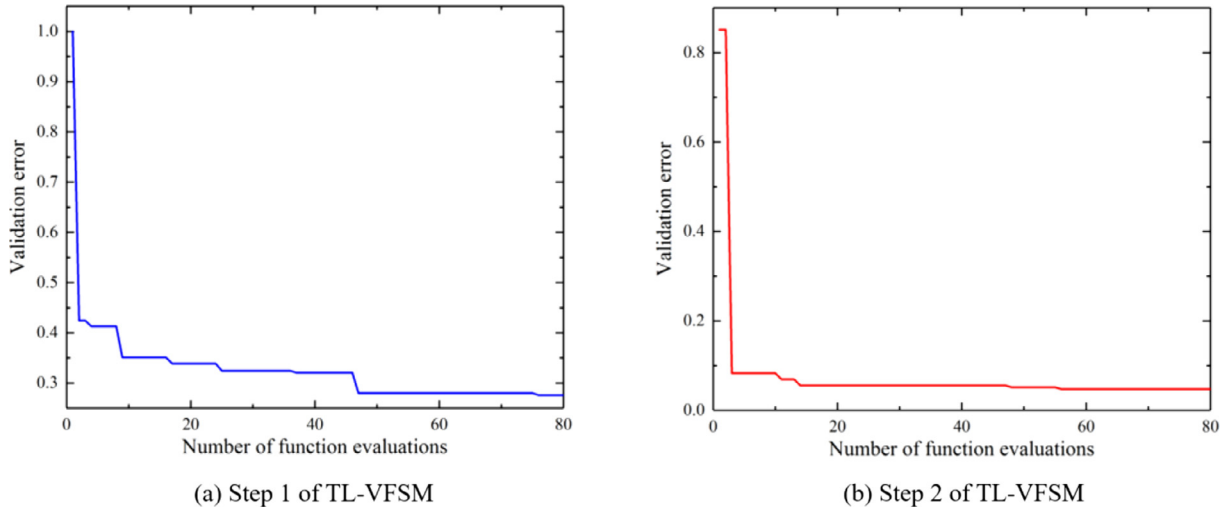

Fig 5. Iteration histories of Bayesian optimizations for TL-VFSM (30HFM + 300LFM) for Convex function.

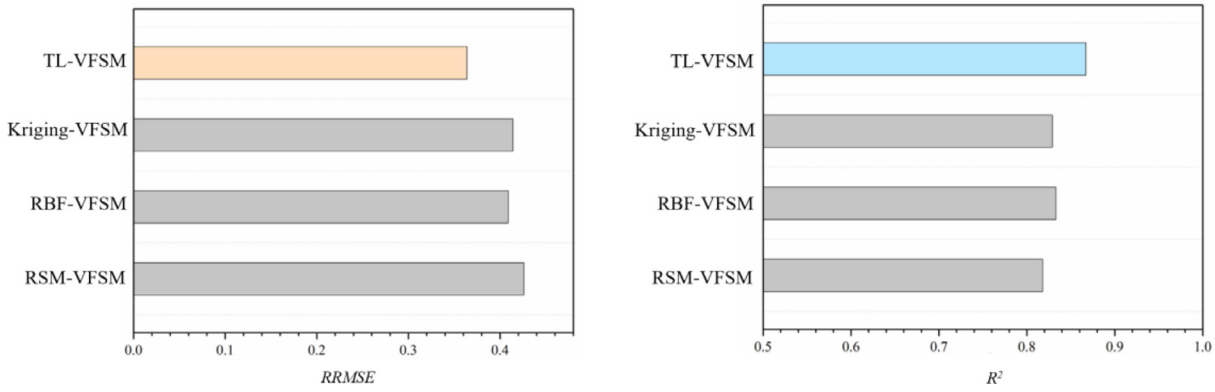

Fig 6. Histograms of prediction accuracy of VFSMs (30HFM + 300LFM) for Convex function.

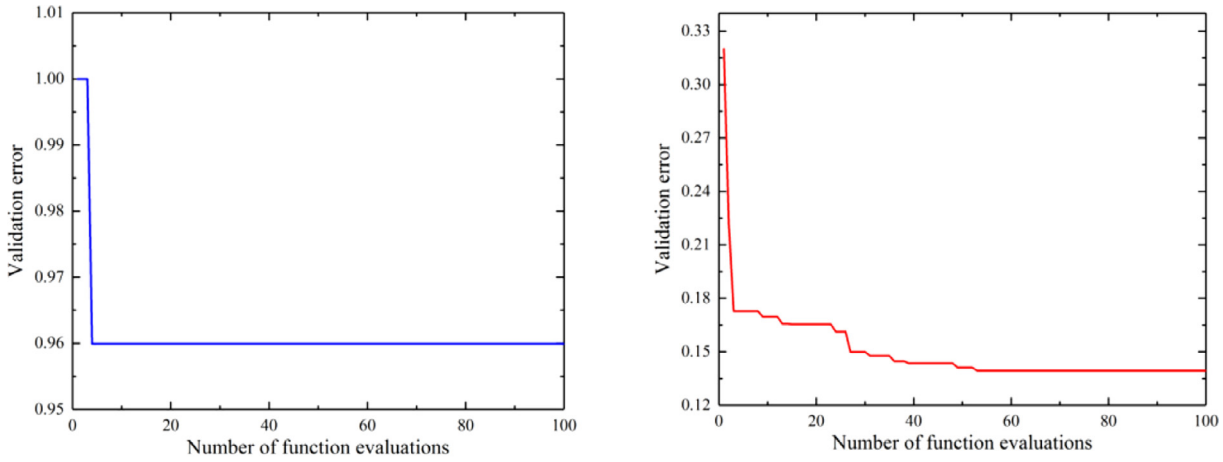

Fig 7. Iteration histories of Bayesian optimizations for TL-VFSM (150HFM + 1500LFM) for Powell function.

Table 8
Hyperparameter results of pre-tuning for TL-VFSM (150HFM + 1500LFM) for Powell function.

Hyperparameters	Lower bound	Tuning result	Upper bound
Number of neurons in each layer	20	110	200
Number of layers	2	17	15
Learning rate	1e−4	6.724e−4	0.1
Decay rate of gradient	0.7	0.977	0.98
Dropout rate	0.2	0.602	0.8
Gradient threshold	0.1	3.175	5
Size of the mini-batch	32	64	128
Factor for L_2 regularization	1e−10	0.0059	1e−2

Table 9
Hyperparameter results of fine-tuning for TL-VFSM (150HFM + 1500LFM) for Powell function.

Hyperparameters	Lower bound	Tuning result	Upper bound
Learning rate	1e−8	5.596e−7	1e−2
Decay rate of gradient	0.7	0.812	0.98
Gradient threshold	0.1	0.229	5
Size of the mini-batch	32	128	128
Factor for L_2 regularization	1e−10	2.521e−6	1e−2
Learning rate factor of weight (the last layer)	1	249	300
Learning rate factor of bias (the last layer)	1	235	300

out in Ref. [55], variable-stiffness composite shells have been used in fuselages of Boeing 787 Dreamliner, A380, etc. In this paper, the buckling problem of variable-stiffness composite shells under bending loading is investigated. Particularly, the example studied in this paper refers to the example mentioned in Ref. [49].

The model information of the variable-stiffness composite shell is introduced as follows. The shell diameter D and the shell length L of the variable-stiffness composite shell are both 457.2 mm. The total number of layers of the variable-stiffness composite shell is 16. The thickness of single layer is 0.127 mm. The material properties are listed in Table 11. The boundary condition of the variable-stiffness composite shell is clamped–clamped at both ends. A bending moment

Table 10
The prediction accuracy of VFSMs (150HFM + 1500LFM) for Powell function.

	RRMSE	R^2
RSM-VFSM	2.272	−4.179
RBF-VFSM	0.615	0.623
Kriging-VFSM	0.650	0.578
TL-VFSM	**0.551**	**0.698**

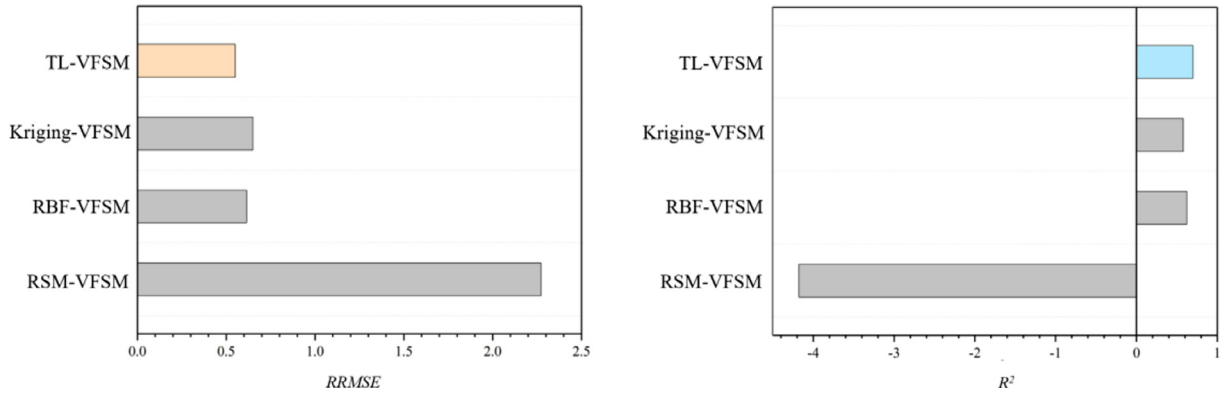

Fig 8. Histograms of prediction accuracy of VFSMs (150HFM + 1500LFM) for Powell function.

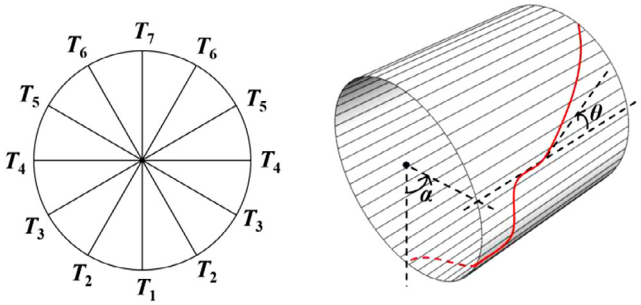

Fig 9. Schematic diagram of design variables and fiber path in each ply of variable-stiffness composite shell (from Ref. [49]).

is imposed on the variable-stiffness composite shell and the linear buckling load is calculated after the eigenvalue buckling analysis. 12 segments are pre-defined along the circumferential direction of the variable-stiffness composite shell, as shown in Fig. 9. Due to the symmetry in loading and boundary conditions of the variable-stiffness composite shell, 7 design variables T_i ($i = 1,\ldots,7$) are used to represent the fiber orientation angles at the boundary of adjacent segments in each ply. The fiber orientation angle $\theta(\alpha)$ varies linearly along the circumferential direction, which is expressed as

$$\theta(\alpha) = T_i + \frac{T_{i+1} - T_i}{\alpha_{i+1} - \alpha_i}(\alpha - \alpha_i) \tag{22}$$

where, α_i and α_{i+1} represent lower and upper bounds of the circumferential azimuth angle α for ith segment. For example, $\alpha_1 = 0$ and $\alpha_2 = \pi/6$ for the first segment. T_i are continuous design variables varying from 0° to 90°. The basic fiber path is shifted along the longitudinal direction to form the entire ply for the variable-stiffness composite shell.

8-node reduced integration shell elements are used for meshing the FE model of variable-stiffness composite shells. The FE models with 6000 elements and 600 elements are regarded as HFM and LFM for variable-stiffness composite shells, respectively, as shown in Fig. 10. A workstation with the CPU of Intel Xeon E5-2687w @ 3.10 GHz and 64G RAM is used. The computational time of HFM and LFM is 190 s and 28 s, respectively. Firstly, the proposed TL-VFSM is used for constructing the VFSM for variable-stiffness composite shells. 300 LFM points and 30 HFM points are sampled using the LHS method and calculated by the eigenvalue analysis method, and their results are then collected into the source task and target task. In Step 1 of TL-VFSM, a DNN model is initialized based on the source task com-

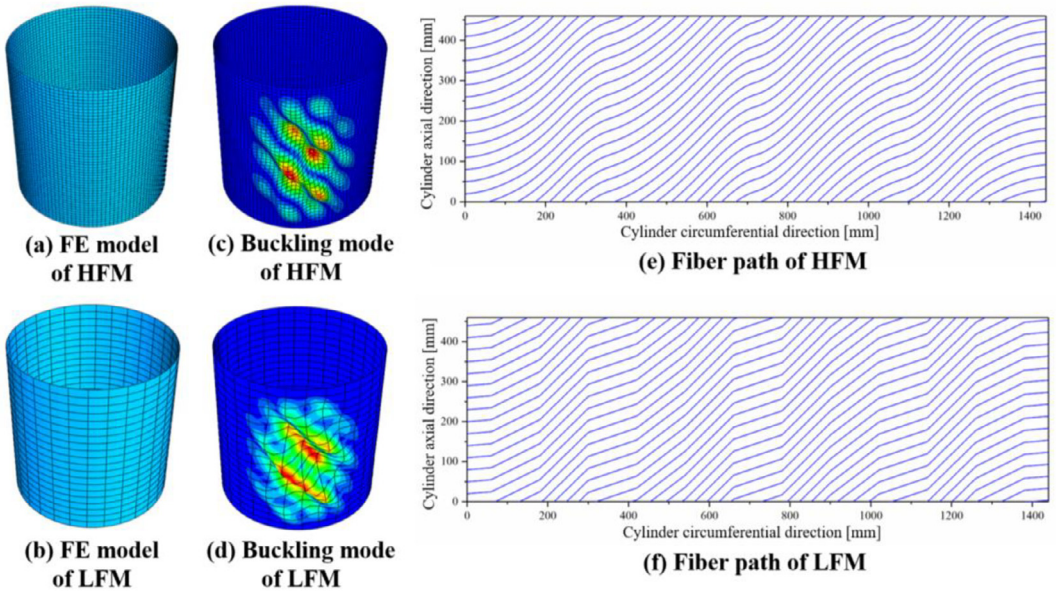

Fig 10. Schematic diagram of HFM and LFM for variable-stiffness composite shells($T_1 = 5$, $T_2 = 30$, $T_3 = 80$, $T_4 = 20$, $T_5 = 50$, $T_6 = 90$, $T_7 = 20$).

Table 11
Material properties of each unidirectional carbon/epoxy composite ply.

E_1[GPa]	E_2[GPa]	E_3[GPa]	G_{12}[GPa]	G_{13}[GPa]	G_{23}[GPa]	ν_{12}	ν_{13}	ν_{23}
134	7.71	7.71	4.31	4.31	2.76	0.301	0.301	0.396

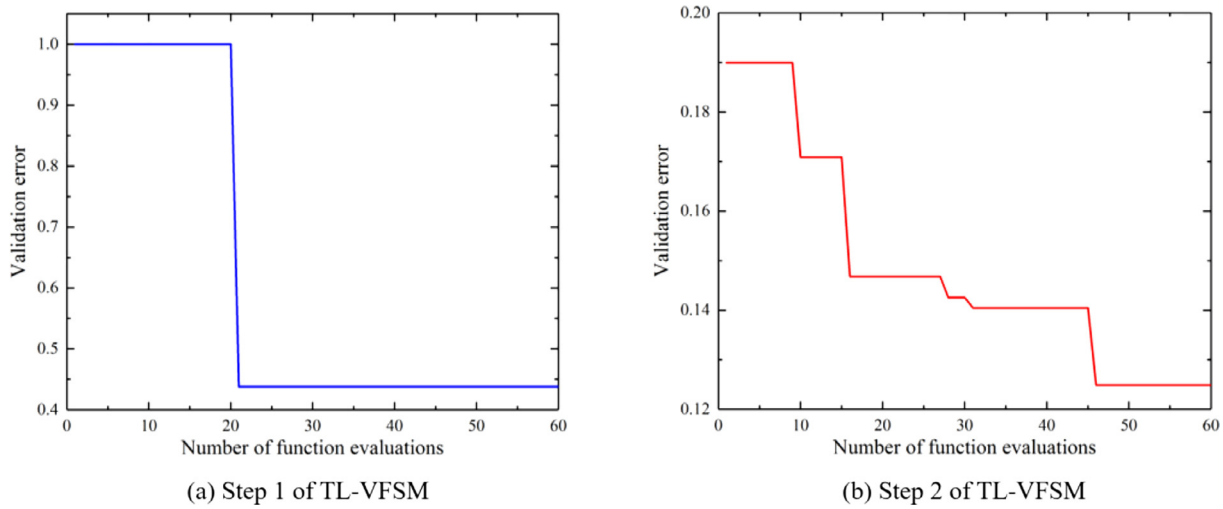

Fig 11. Iteration histories of Bayesian optimizations for TL-VFSM (30HFM + 300LFM) for variable-stiffness composite shells.

Table 12
Hyperparameter results of pre-tuning for TL-VFSM (30HFM + 300LFM) for variable-stiffness composite shells.

Hyperparameters	Lower bound	Tuning result	Upper bound
Number of neurons in each layer	20	134	200
Number of layers	2	3	15
Learning rate	1e−4	0.0665	0.1
Decay rate of gradient	0.7	0.895	0.98
Dropout rate	0.2	0.668	0.8
Gradient threshold	0.1	0.356	5
Size of the mini-batch	32	64	128
Factor for L_2 regularization	1e−10	1.889e−5	1e−2

Table 13
Hyperparameter results of fine-tuning for TL-VFSM (30HFM + 300LFM) for variable-stiffness composite shells.

Hyperparameters	Lower bound	Tuning result	Upper bound
Learning rate	1e−8	2.578e−5	1e−2
Decay rate of gradient	0.7	0.904	0.98
Gradient threshold	0.1	2.307	5
Size of the mini-batch	32	128	128
Factor for L_2 regularization	1e−10	0.0031	1e−2
Learning rate factor of weight (the last layer)	1	193	300
Learning rate factor of bias (the last layer)	1	12	300

Table 14
The prediction accuracy of VFSMs (30HFM + 300LFM) for variable-stiffness composite shells.

	RRMSE	R^2
RSM-VFSM	0.426	0.818
RBF-VFSM	0.409	0.833
Kriging-VFSM	0.414	0.829
TL-VFSM	**0.364**	**0.867**

posed of 300 LFM points. The hyperparameters of the DNN model are tuned using BO on the target task. After the pre-tuning, the pre-trained DNN model is obtained. The maximum iteration number N of BO is 60. The iteration history of BO is displayed in Fig. 11. The normalization factor for validation error in Fig. 11 is 720. The optimized results of hyperparameters of the pre-tuning are listed in Table 12. In Step 2 of TL-VFSM, early layers of the pre-trained DNN model are remained, while the last layer of the pre-trained DNN model is replaced. Then, another BO hyperparameter tuning is carried out to fine-tune the pre-trained DNN model on the target task. The iteration history of BO is displayed in Fig. 11. The optimized results of hyperparameters of the fine-tuning are shown in Table 13. After above two steps, the TL-VFSM is constructed. The prediction accuracy results of the TL-VFSM tested by 100 HFM points are listed in Table 14 and Fig. 12. In addition, the RSM-VFSM, RBF-VFSM and Kriging-VFSM are also constructed using the same number of HFM and LFM points as TL-VFSM. The prediction accuracy results of RSM-VFSM, RBF-VFSM and Kriging-VFSM are listed in Table 14 and Fig. 12. It can be observed that, the prediction error result (*RRMSE* result) of TL-VFSM decreases by 15%, 11% and 12% than prediction error results of RSM-VFSM, RBF-VFSM and Kriging-VFSM, respectively, which indicates that the proposed TL-VFSM achieves higher prediction accuracy than traditional VFSMs.

Moreover, a comparison is made from the view-of-point of the computational cost. The HFSMs are constructed based on 30, 60, 90, 120, 150 and 180 HFM points, and their prediction accuracy results are calculated, respectively. As displayed in Fig. 13, the prediction error result (*RRMSE* result) of HFSMs decreases gradually as the number of HFM points increases. It can be found that the prediction error result of the HFSM (180HFM) is close to that of the TL-VFSM (30HFM + 300LFM). As for the total computational time, the TL-VFSM (30HFM + 300LFM) needs about 14,100 s, which decreases by 59% than the HFSM (180HFM) (34,200 s), indicating the high computational efficiency of the proposed method. Above all, the effectiveness of the proposed TL-VFSM has been verified for the engineering example of variable-stiffness composite shells with 7 continuous variables.

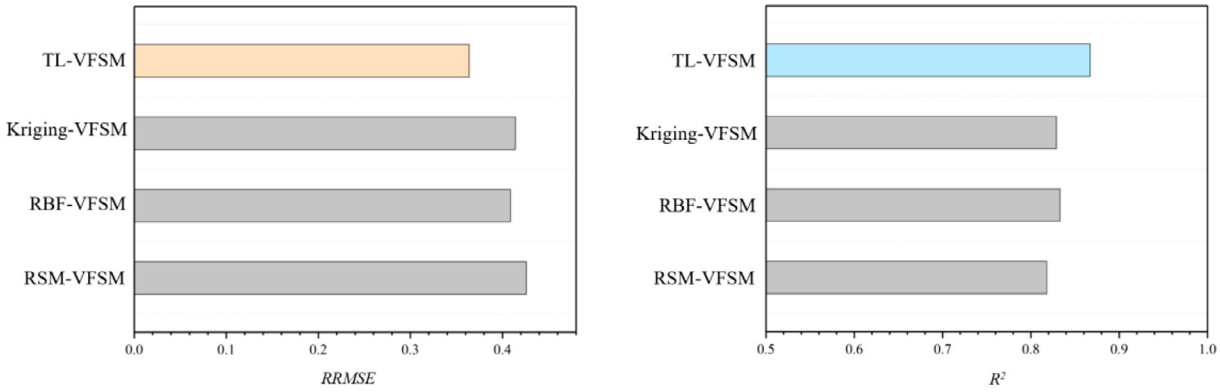

Fig 12. Histograms of prediction accuracy of VFSMs (30HFM + 300LFM) for variable-stiffness composite shells.

	30 HFM	60 HFM	90 HFM	120 HFM	150 HFM	180 HFM
RRMSE	0.813	0.630	0.470	0.452	0.416	0.385
R^2	0.333	0.599	0.777	0.793	0.825	0.850

Fig 13. The prediction accuracy of HFSMs for variable-stiffness composite shells.

3.5. Engineering example 2: Hierarchical stiffened shells (9 design variables)

Next, another engineering example of the hierarchical stiffened shell is studied. The hierarchical stiffened shell is an innovative load-carrying thin-walled structure used in interstages of launch vehicles in the aerospace field [12,18,56–60]. In comparison to traditional stiffened shells, hierarchical stiffened shells have higher load-carrying capacity [12]. In order to accurately capture the pre-buckling path, the collapse point and the post-buckling path, the explicit dynamics method has been widely used for stiffened shells and hierarchical stiffened shells, and it was validated to have a good agreement with experimental results [61]. Thus, the FE model with full structural details calculated by the explicit dynamic method is considered as the HFM for hierarchical stiffened shells, as shown in Fig. 14(a). However, the HFM is generally time-consuming because a large number of FE meshes are needed to simulate the complex stiffeners of hierarchical stiffened shells, especially the minor stiffeners [12]. It can be expected that, if the minor stiffeners were smeared out, the computational cost of hierarchical stiffened shells would decrease significantly. The Numerical Implementation of Asymptotic Homogenization (NIAH) method is employed for the equivalence of minor stiffeners, and thus the hybrid model can be established as the LFM for hierarchical stiffened shells, as shown in Fig. 14(b). More details about the establishment methods of HFM and LFM of hierarchical stiffened shells can refer to Ref. [12]. Alternative establishment methods of LFM for hierarchical stiffened shells might be reduced-order methods [62].

The model information of the hierarchical stiffened shell is as follows. As listed in Table 15, the hierarchical stiffened shell has a diameter D of 3000 mm and a length L of 2000 mm. The mechanical properties of the hierarchical stiffened shell are as follows: Young's modulus $E = 70.0$ GPa and Poisson's ratio $\upsilon = 0.33$, yield stress $\sigma_s = 563$ MPa, ultimate stress $\sigma_b = 630$ MPa and elongation $\delta = 0.07$. The hierarchical stiffened shell contains nine design variables (five continuous variables and four discrete variables) as follows. t_s is the skin thickness, N_{aj} is the number of axial major stiffeners, N_{an} is the axial minor stiffeners between axial major stiffeners, N_{cj} is the number of circumferential major stiffeners, N_{cn} is the circumferential minor stiffeners between circumferential major stiffeners, h_{rj} and h_{rn} are the major and minor stiffener heights respectively, t_{rj} and t_{rn} are the major and minor stiffener thicknesses. The design space of design variables are listed in Table 16. The boundary condition is to keep the lower end of the hierarchical stiffened shell clamped and the upper end fixed except the degrees of freedom along the axial direction. A uniform axial load is applied to the upper end of the hierarchical stiffened shell. For single calculation, HFM and LFM of hierarchical stiffened shells need about 1.5 h and 0.08 h, respectively. In other words, the computational time of 19 LFM equals to that of 1 HFM. The calculation results of HFM and LFM are shown in Fig. 14(c).

Table 15
The dimension and material information of hierarchical stiffened shells.

D [mm]	L [mm]	E [MPa]	υ	σ_s [MPa]	σ_b [MPa]	δ
3000	2000	70,000	0.33	563	630	0.07

Table 16
The design space of hierarchical stiffened shells.

	t_s [mm]	t_{rj} [mm]	t_{rn} [mm]	h_j [mm]	h_n [mm]	N_{cj}	N_{cn}	N_{aj}	N_{an}
Lower bound	2.5	3.0	3.0	15.0	6.0	3	1	20	1
Upper bound	5.5	12.0	12.0	30.0	15.0	9	4	50	4

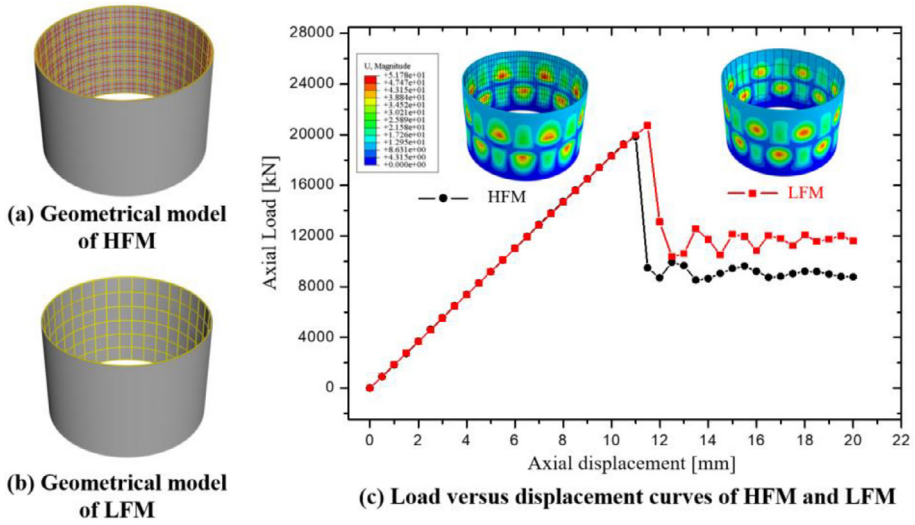

Fig 14. Schematic diagram of HFM and LFM for hierarchical stiffened shells(t_s = 4.3 mm, t_{rj} = 3.2 mm, t_{rn} = 8.2 mm, h_j = 30.0 mm, h_n = 13.9 mm, N_{cj} = 5, N_{cn} = 2, N_{aj} = 39, N_{an} = 3).

Table 17
Hyperparameter results of pre-tuning for TL-VFSM (30HFM + 200LFM) for hierarchical stiffened shells.

Hyperparameters	Lower bound	Tuning result	Upper bound
Number of neurons in each layer	20	179	200
Number of layers	2	10	15
Learning rate	1e−4	2.311e−4	0.1
Decay rate of gradient	0.7	0.953	0.98
Dropout rate	0.2	0.393	0.8
Gradient threshold	0.1	1.394	5
Size of the mini-batch	32	64	128
Factor for L_2 regularization	1e−10	4.211e−4	1e−2

Table 18
Hyperparameter results of fine-tuning for TL-VFSM (30HFM + 200LFM) for hierarchical stiffened shells.

Hyperparameters	Lower bound	Tuning result	Upper bound
Learning rate	1e−8	2.110e−5	1e−2
Decay rate of gradient	0.7	0.889	0.98
Gradient threshold	0.1	0.951	5
Size of the mini-batch	32	32	128
Factor for L_2 regularization	1e−10	8.85e−4	1e−2
Learning rate factor of weight (the last layer)	1	38	300
Learning rate factor of bias (the last layer)	1	39	300

Firstly, the effectiveness of the proposed TL-VFSM is investigated. 200 LFM points and 30 HFM points are sampled by the LHS method and calculated, respectively, which are then collected into the source task and target task. The maximum iteration number N of BO is 80. The iteration history of BO is displayed in Fig. 15 and the normalization factor for validation error is 1.09e + 8. The optimized results of hyperparameters of the pre-tuning are listed in Table 17. The iteration history of BO is displayed in Fig. 15. The optimized results of hyperparameters of the fine-tuning are shown in Table 18. After the transfer learning, the TL-VFSM is constructed. Another set of 200 HFM points are generated as the test set for TL-VFSM. The prediction accuracy result of the TL-VFSM is listed in Table 19 and Fig. 16. In order to make a comparison, the RSM-VFSM, RBF-VFSM and Kriging-VFSM are constructed and their prediction accuracy results are also listed in Table 19 and Fig. 16. By contrast, the prediction error result (*RRMSE* result) of TL-VFSM decreases by 51%, 15% and 25% than prediction error results of RSM-VFSM, RBF-VFSM and Kriging-VFSM, respectively. Then, the computational time for training one VFSM and one TL-VFSM is compared, as shown in Table 19. If the RSM method and the additive bridge function are used to build the VFSM, it only needs 0.183 s. This process is quite quick because it is an interpolation process rather than a training of machine learning. If TL-VFSM is trained without BO hyperparameter tuning, it needs about 40 s. If TL-VFSM is trained using BO hyperparameter tuning, it needs about 5271 s. The training time of TL-VFSM is optional according to the requirement of users. If the user does not care much about the performance, it can train the TL-VFSM without BO hyperparameter tuning. In this case, the training time of the VFSM and the TL-VFSM is quite close. If the user wants to obtain an outstanding performance, it can train the TL-VFSM using BO hyperparameter tuning. In this case, the training time is a little longer. But it is quite close to the single computational time of one HFM data (about 5400 s), so the training time seems to be not that longer. By contrast, the total time for training the VFSM or TL-VFSM is far smaller than the total time (219600 s) for generating and calculating the sample data set. Since our aim is to obtain the best performance (prediction accuracy) of TL-VFSM, the training time of TL-VFSM is acceptable.

For comparison, the prediction accuracy of the HFSM based on HFM points is investigated. We increase the number of HFM points

Table 19
The prediction accuracy of VFSMs (30HFM + 200LFM) for hierarchical stiffened shells.

	Computational time of sampling data [s]	Computational time of training [s]	*RRMSE*	R^2
RSM-VFSM	219,600	0.183	0.455	0.794
RBF-VFSM	219,600	0.00127	0.262	0.932
Kriging-VFSM	219,600	0.401	0.297	0.912
TL-VFSM	**219,600**	**5271**	**0.223**	**0.950**

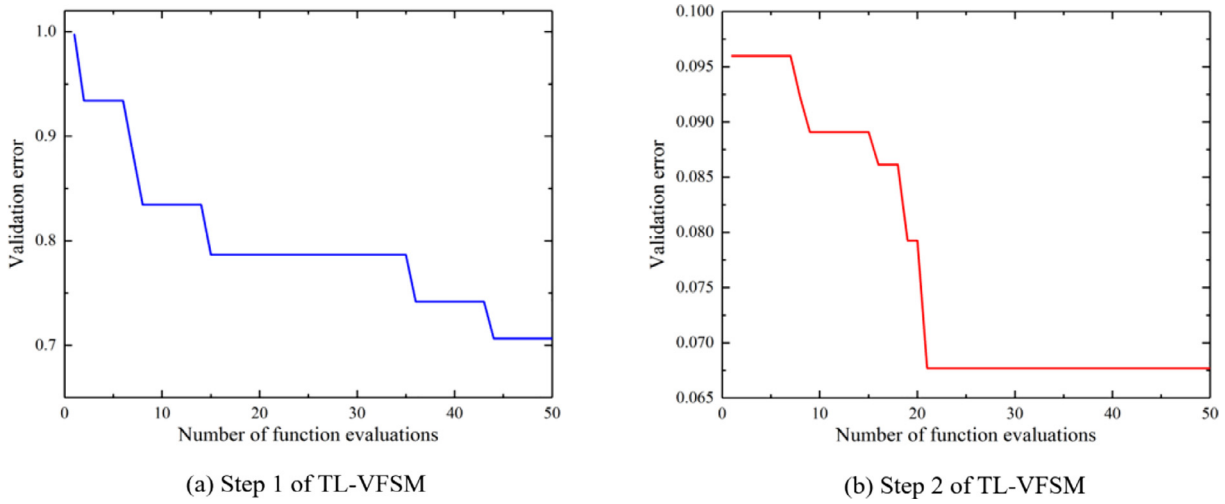

Fig 15. Iteration histories of Bayesian optimizations for TL-VFSM (30HFM + 200LFM) for hierarchical stiffened shells.

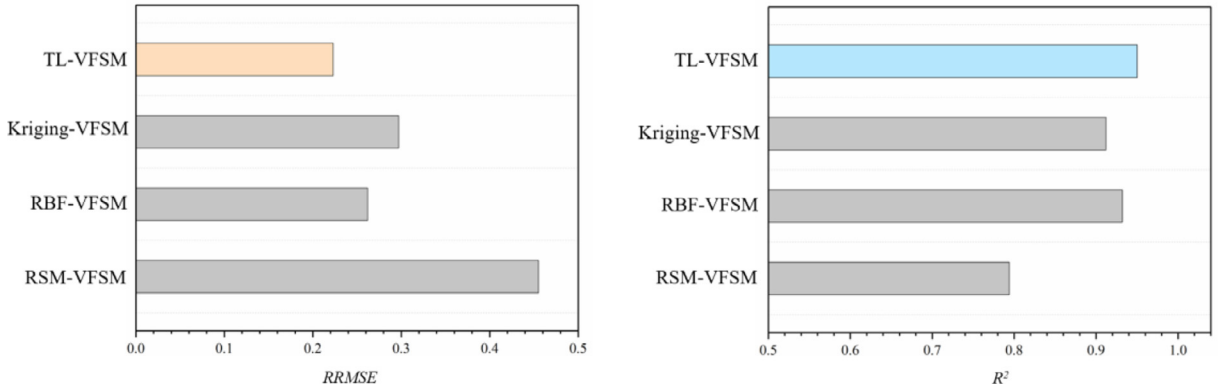

Fig 16. Histograms of prediction accuracy of VFSMs (30HFM + 200LFM) for hierarchical stiffened shells.

Table 20
The prediction accuracy of HFSMs for hierarchical stiffened shells.

	10 HFM	15 HFM	20 HFM	30 HFM	40 HFM	60 HFM	80 HFM
Computational time of sampling data [s]	54,000	81,000	108,000	162,000	216,000	324,000	432,000
Computational time of training [s]	0.00917	0.00307	0.00142	0.00146	0.00615	0.00237	0.00157
RRMSE	0.515	0.396	0.335	0.302	0.274	0.247	0.220
R^2	0.733	0.842	0.887	0.909	0.925	0.936	0.951

used for constructing the HFSM from 10 to 80. The prediction accuracy results of HFSMs are shown in Table 20. It can be found that, as the number of HFM points increases, the prediction error result (*RRMSE* result) decreases gradually. The prediction error result of HFSM (80HFM) is close to that of the TL-VFSM (30HFM + 200LFM). As we mentioned previously, the computational time of 19 LFM points equals to that of 1 HFM point. That is to say, the total computational cost of the TL-VFSM (30HFM + 200LFM) decreases by 49% than that of the HFSM (80HFM) when they achieve similar prediction accuracy. Through the engineering example of hierarchical stiffened shells, the high prediction accuracy and efficiency of the proposed TL-VFSM has been fully demonstrated.

4. Conclusions

The TL-VFSM is introduced in this paper to provide a novel construction method of VFSM without the bridge function. The main idea of the TL-VFSM is to take full advantage of the strong training ability of the transfer learning method to learn the knowledge from the source dataset composed of LFM points and then improve the learning of the target dataset composed of HFM points. The proposed TL-VFSM can be divided into two steps. In Step 1, a DNN model is initialized using the source dataset. Then, the hyperparameters of the DNN model are pre-tuned using BO and the optimization objective of BO is to minimize the validation error of the DNN model. After that, the pre-trained DNN model is obtained. In Step 2, early layers of the pre-trained DNN are remained while the last layer of the pre-trained DNN is re-initialized. Then, another hyperparameter tuning is performed to fine-tune the pre-trained DNN using BO. After this process, the re-trained DNN model is obtained. Through above two steps, the TL-VFSM is constructed.

The Dette & Pepelyshev function (8 dimensions), the Convex function (8 dimensions), the Powell function (16 dimensions), the engineering example of variable-stiffness composite shells (7 design

variables) and the engineering example of hierarchical stiffened shells (9 design variables) are employed to verify the effectiveness of the proposed TL-VFSM, respectively. Example results indicate that using the same number of HFM and LFM points, the proposed TL-VFSM achieves higher prediction accuracy than the traditional VFSM constructed by the common bridge functions. If we do not know which bridge function is appropriate for the complex non-linear relationship between the HFM data and the LFM data, we need to try multiplicative bridge function, additive bridge function, comprehensive bridge function or even learn other new bridge functions. This process need us to learn the theory of a new bridge function if all available bridge functions are not effective, write codes for the new bridge function and so on. In addition, whether the used bridge function is effective and accurate is case by case. By contrast, TL-VFSM is more flexible and robust because it can obtain better performance by the automatic training and learning. In future work, we will carry out the training by GPU computing, which can significantly increase the training efficiency by about 20 times. Through the examples of the Dette & Pepelyshev function and the Convex function, the effectiveness of the proposed TL-VFSM is fully verified in predicting the continuous variable problem and the mixed-integer variable problem, respectively. As for the engineering examples of variable-stiffness composite shells and hierarchical stiffened shells, they are both related to the buckling problems of thin-walled shells. It can be expected that the proposed TL-VFSM would have a promising engineering application value and prospectively be applied in the design of fuel tanks and interstages widely used in the aerospace field. In the future study, the proposed TL-VFSM would be extended to variable-fidelity surrogate optimizations.

CRediT authorship contribution statement

Kuo Tian: Conceptualization, Methodology, Writing - original draft. **Zengcong Li:** Methodology, Software, Validation. **Jiaxin Zhang:** Writing- Reviewing and Editing. **Jiaxin Zhang:** . **Lei Huang:** Methodology, Software, Validation. **Bo Wang:** Supervision, Writing - review & editing, Project administration.

Declaration of Competing Interest

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

Acknowledgments

This work was supported by National Natural Science Foundation of China [No. 11902065, No. 11825202], China Postdoctoral Science Foundation [No. 2019M651107], LiaoNing Revitalization Talents Program [No. XLYC1802020] and the Fundamental Research Funds for the Central Universities [No. DUT21RC(3)013].

Data availability

The raw/processed data required to reproduce these findings cannot be shared at this time as the data also forms part of an ongoing study.

References

[1] Jouhaud J-C, Sagaut P, Montagnac M, Laurenceau J. A surrogate-model based multidisciplinary shape optimization method with application to a 2D subsonic airfoil. Comput Fluids 2007;36(3):520–9.
[2] Zhang M, Gou W, Li L, Yang F, Yue Z. Multidisciplinary design and multi-objective optimization on guide fins of twin-web disk using Kriging surrogate model. Struct Multidiscip Optim 2017;55(1):361–73.
[3] Rikards R, Abramovich H, Kalnins K, Auzins J. Surrogate modeling in design optimization of stiffened composite shells. Compos Struct 2006;73(2):244–51.
[4] Arian Nik M, Fayazbakhsh K, Pasini D, Lessard L. Surrogate-based multi-objective optimization of a composite laminate with curvilinear fibers. Compos Struct 2012;94(8):2306–13.
[5] Arian Nik M, Fayazbakhsh K, Pasini D, Lessard L. A comparative study of metamodeling methods for the design optimization of variable stiffness composites. Compos Struct 2014;107:494–501.
[6] Meng Z, Zhang Z, Zhang D, Yang D. An active learning method combining Kriging and accelerated chaotic single loop approach (AK-ACSLA) for reliability-based design optimization. Comput Methods Appl Mech Eng 2019;357:112570. https://doi.org/10.1016/j.cma.2019.112570.
[7] Meng Z, Zhang Z, Li G, et al. An active weight learning method for efficient reliability assessment with small failure probability. Struct Multidiscip Optim 2019:1–14.
[8] Meng Z, Zhang D, Li G, Yu Bo. An importance learning method for non-probabilistic reliability analysis and optimization. Struct Multidiscip Optim 2019;59(4):1255–71.
[9] Wang D, Yeo S-Y, Su Z, Wang Z-P, Abdalla MM. Data-driven streamline stiffener path optimization (SSPO) for sparse stiffener layout design of non-uniform curved grid-stiffened composite (NCGC) structures. Comput Methods Appl Mech Eng 2020;365:113001. https://doi.org/10.1016/j.cma.2020.113001.
[10] Wagner HNR, Köke H, Dähne S, Niemann S, Hühne C, Khakimova R. Decision tree-based machine learning to optimize the laminate stacking of composite cylinders for maximum buckling load and minimum imperfection sensitivity. Compos Struct 2019;220:45–63.
[11] Tian K, Li Z, Huang L, Du K, Jiang L, Wang Bo. Enhanced variable-fidelity surrogate-based optimization framework by Gaussian process regression and fuzzy clustering. Comput Methods Appl Mech Eng 2020;366:113045. https://doi.org/10.1016/j.cma.2020.113045.
[12] Tian K, Wang Bo, Zhang Ke, Zhang J, Hao P, Wu Y. Tailoring the optimal load-carrying efficiency of hierarchical stiffened shells by competitive sampling. Thin-Walled Struct 2018;133:216–25.
[13] Forrester AIJ, Sóbester A, Keane AJ. Multi-fidelity optimization via surrogate modelling. Proc Royal Soc A: Math Phys Eng Sci 2007;463(2088):3251–69.
[14] Han Z-H, Görtz S. Hierarchical kriging model for variable-fidelity surrogate modeling. AIAA J 2012;50(9):1885–96.
[15] Park C, Haftka RT, Kim NH. Low-fidelity scale factor improves Bayesian multi-fidelity prediction by reducing bumpiness of discrepancy function. Struct Multidiscip Optim 2018;58(2):399–414.
[16] Giselle Fernández-Godino M, Park C, Kim NH, Haftka RT. Issues in deciding whether to use multifidelity surrogates. AIAA J 2019;57(5):2039–54.
[17] Han Z, Xu C, Zhang L, Zhang Yu, Zhang K, Song W. Efficient aerodynamic shape optimization using variable-fidelity surrogate models and multilevel computational grids. Chin J Aeronaut 2020;33(1):31–47.
[18] Tian K, Li Z, Ma X, Zhao H, Zhang J, Wang Bo. Toward the robust establishment of variable-fidelity surrogate models for hierarchical stiffened shells by two-step adaptive updating approach. Struct Multidiscip Optim 2020;61(4):1515–28.
[19] Peherstorfer B, Willcox K, Gunzburger M. Survey of multifidelity methods in uncertainty propagation, inference, and optimization. SIAM Rev 2018;60(3):550–91.
[20] Song X, Lv L, Sun W, Zhang J. A radial basis function-based multi-fidelity surrogate model: exploring correlation between high-fidelity and low-fidelity models. Struct Multidiscip Optim 2019;60(3):965–81.
[21] Han Z, Zimmerman R, Görtz S. Alternative cokriging method for variable-fidelity surrogate modeling. AIAA J 2012;50(5):1205–10.
[22] Han Z-H, Görtz S, Zimmermann R. Improving variable-fidelity surrogate modeling via gradient-enhanced kriging and a generalized hybrid bridge function. Aerosp Sci Technol 2013;25(1):177–89.
[23] Zhou Qi, Shao X, Jiang P, Zhou H, Shu L. An adaptive global variable fidelity metamodeling strategy using a support vector regression based scaling function. Simul Model Pract Theory 2015;59:18–35.
[24] Fernández-Godino M G, Park C, Kim N H, et al. Review of multi-fidelity models. arXiv preprint arXiv:1609.07196, 2016.
[25] Dong H, Song B, Wang P, Huang S. Multi-fidelity information fusion based on prediction of kriging. Struct Multidiscip Optim 2015;51(6):1267–80.
[26] Zhou Qi, Yang Y, Jiang P, Shao X, Cao L, Hu J, et al. A multi-fidelity information fusion metamodeling assisted laser beam welding process parameter optimization approach. Adv Eng Softw 2017;110:85–97.
[27] Zhou Qi, Shao X, Jiang P, Gao Z, Wang C, Shu L. An active learning metamodeling approach by sequentially exploiting difference information from variable-fidelity models. Adv Eng Inf 2016;30(3):283–97.
[28] Tao J, Sun G. Application of deep learning based multi-fidelity surrogate model to robust aerodynamic design optimization. Aerosp Sci Technol 2019;92:722–37.
[29] Pilania G, Gubernatis JE, Lookman T. Multi-fidelity machine learning models for accurate bandgap predictions of solids. Comput Mater Sci 2017;129:156–63.
[30] Liu H, Ong Y-S, Cai J, Wang Yi. Cope with diverse data structures in multi-fidelity modeling: a Gaussian process method. Eng Appl Artif Intell 2018;67:211–25.
[31] Pan SJ, Yang Q. A survey on transfer learning. IEEE Trans Knowl Data Eng 2009;22(10):1345–59.
[32] Goodfellow I, Bengio Y, Courville A. Deep Learning. MIT Press; 2016.
[33] X. Glorot, A. Bordes, Y. Bengio, Deep sparse rectifier neural networks, in: Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, pp. 315–323.
[34] Courbariaux M, Bengio Y, David J P. Binaryconnect: Training deep neural networks with binary weights during propagations. Advances in neural information processing systems. 2015: 3123-3131.

[35] Kingma, Diederik, and Jimmy Ba. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014).
[36] Mockus J, Tiesis V, Zilinskas A. The application of Bayesian methods for seeking the extremum. Towards Global Optim 1978;2(117–129):2.
[37] Zhang J, Liu X, Bi S, Yin J, Zhang G, Eisenbach M. Robust data-driven approach for predicting the configurational energy of high entropy alloys. Mater Des 2020;185:108247. https://doi.org/10.1016/j.matdes.2019.108247.
[38] Jang H, Plis SM, Calhoun VD, Lee J-H. Task-specific feature extraction and classification of fMRI volumes using a deep neural network initialized with a deep belief network: Evaluation using sensorimotor tasks. NeuroImage 2017;145:314–28.
[39] Kim J, Calhoun VD, Shim E, Lee J-H. Deep neural network with weight sparsity control and pre-training extracts hierarchical features and enhances classification performance: Evidence from whole-brain resting-state functional connectivity patterns of schizophrenia. NeuroImage 2016;124:127–46.
[40] Lu J, Behbood V, Hao P, Zuo H, Xue S, Zhang G. Transfer learning using computational intelligence: a survey. Knowl-Based Syst 2015;80:14–23.
[41] Cireşan DC, Meier U, Schmidhuber J. Transfer learning for Latin and Chinese characters with deep neural networks. The, international joint conference on neural networks (IJCNN). IEEE 2012;2012:1–6.
[42] Dette H, Pepelyshev A. Generalized Latin hypercube design for computer experiments. Technometrics 2010;52(4):421–9.
[43] Müller J, Shoemaker CA, Piché R. SO-MI: A surrogate model algorithm for computationally expensive nonlinear mixed-integer black-box global optimization problems. Comput Oper Res 2013;40(5):1383–400.
[44] Laguna M, Martí R. Experimental testing of advanced scatter search designs for global optimization of multimodal functions. J Global Optim 2005;33(2):235–55.
[45] Nguyen MH, Vijayachandran AA, Davidson P, Call D, Lee D, Waas AM. Effect of automated fiber placement (AFP) manufacturing signature on mechanical performance of composite structures. Compos Struct 2019;228:111335. https://doi.org/10.1016/j.compstruct.2019.111335.
[46] Hyer MW, Charette RF. Use of curvilinear fiber format in composite structure design. AIAA J 1991;29(6):1011–5.
[47] Abdalla MM, Setoodeh S, Gürdal Za. Design of variable stiffness composite panels for maximum fundamental frequency using lamination parameters. Compos Struct 2007;81(2):283–91.
[48] Lopes CS, Gürdal Z, Camanho PP. Tailoring for strength of composite steered-fibre panels with cutouts. Compos A Appl Sci Manuf 2010;41(12):1760–7.
[49] Rouhi M, Ghayoor H, Hoa SV, Hojjati M. Effect of structural parameters on design of variable-stiffness composite cylinders made by fiber steering. Compos Struct 2014;118:472–81.
[50] Ma X, Tian K, Li H, Zhou Y, Hao P, Wang B. Concurrent multi-scale optimization of hybrid composite plates and shells for vibration. Compos Struct 2020;233:111635. https://doi.org/10.1016/j.compstruct.2019.111635.
[51] Hao P, Liu C, Yuan X, Wang B, Li G, Zhu T, et al. Buckling optimization of variable-stiffness composite panels based on flow field function. Compos Struct 2017;181:240–55.
[52] Wang D, Abdalla MM, Zhang W. Sensitivity analysis for optimization design of non-uniform curved grid-stiffened composite (NCGC) structures. Compos Struct 2018;193:224–36.
[53] Li E. Fast cylinder variable-stiffness design by using Kriging-based hybrid aggressive space mapping method. Adv Eng Softw 2017;114:215–26.
[54] White SC, Weaver PM, Wu KC. Post-buckling analyses of variable-stiffness composite cylinders in axial compression. Compos Struct 2015;123:190–203.
[55] Blom AW. Structural performance of fiber-placed variable-stiffness composite conical and cylindrical shells. Delft University of Technology; 2010.
[56] Li M, Sun F, Lai C, Fan H, Ji B, Zhang X, et al. Fabrication and testing of composite hierarchical isogrid stiffened cylinder. Compos Sci Technol 2018;157:152–9.
[57] Zhang B, Jin F, Zhao Z, Zhou Z, Xu Y, Chen H, et al. Hierarchical anisogrid stiffened composite panel subjected to blast loading: Equivalent theory. Compos Struct 2018;187:259–68.
[58] Sim C-H, Park J-S, Kim H-Il, Lee Y-L, Lee K. Postbuckling analyses and derivations of knockdown factors for hybrid-grid stiffened cylinders. Aerosp Sci Technol 2018;82-83:20–31.
[59] Zhao Y, Chen M, Yang F, Zhang L, Fang D. Optimal design of hierarchical grid-stiffened cylindrical shell structures based on linear buckling and nonlinear collapse analyses. Thin-Walled Struct. 2017;119:315–23.
[60] Wang B, Tian K, Zhou C, Hao P, Zheng Y, Ma Y, et al. Grid-pattern optimization framework of novel hierarchical stiffened shells allowing for imperfection sensitivity. Aerosp Sci Technol 2017;62:114–21.
[61] Tian K, Wang B, Hao P, Waas AM. A high-fidelity approximate model for determining lower-bound buckling loads for stiffened shells. Int J Solids Struct 2018;148-149:14–23.
[62] Liang K, Sun Q. Reduced-order modeling analysis of shell structures buckling using a co-rotational solid-shell element. Aerosp Sci Technol 2017;70:435–44.