Composites Science and Technology 224 (2022) 109425

Contents lists available at ScienceDirect

Composites Science and Technology

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

Stochastic integrated machine learning based multiscale approach for the prediction of the thermal conductivity in carbon nanotube reinforced polymeric composites

Bokai Liu [a], Nam Vu-Bac [b], Xiaoying Zhuang [b], Xiaolong Fu [c], Timon Rabczuk [a,*]

[a] Institute of Structural Mechanics, Bauhaus-Universität Weimar, Marienstr. 15, D-99423, Weimar, Germany
[b] Institute of Photonics, Gottfried Wilhelm Leibniz Universität Hannover, 30167, Hannover, Germany
[c] Xi'an Modern Chemistry Research Institute, Xi'an, 710065, China

ARTICLE INFO

Keywords:
Carbon nanotube reinforced polymeric composites (CNT-PCs)
Machine learning
Multi-scale stochastic modeling
Thermal properties
Computational complexity

ABSTRACT

We present a stochastic integrated machine learning based multiscale approach for the prediction of the macroscopic thermal conductivity in carbon nanotube reinforced polymeric composites (CNT-PCs). Seven types of machine learning models are exploited, namely Multivariate Adaptive Regression Splines (MARS), Support Vector Machine (SVM), Regression Tree (RT), Bagging Tree (Bag), Random Forest (RF), Gradient Boosting Machine (GBM) and Cubist. They are used as components of stochastic modeling constructing the relationship between all uncertain inputs variables and the output of interest, the macroscopic thermal conductivity of the composite. Particle Swarm Optimization (PSO) is used for hyper-parameter tuning to find the global optimal values leading to a significant reduction in the computational cost. We also analyze the advantages and disadvantages of various methods in terms of computational expense and model complexity. We believe that the presented stochastic integrated machine learning approach accounting for uncertainties is a valuable step towards computational design of new composites for application related to thermal management.

1. Introduction

Carbon nanotube reinforced polymeric composites (CNT-PCs) have been studied and developed due to their excellent physical and chemical properties [1,2]. Many studies in recent years have focused on quantifying the effect of fillers on the performance of those composites. One material property is the macroscopic thermal conductivity which is important in many engineering applications including aerospace engineering, the automotive industry, energy storage equipment and electronic devices, to name a few. For devices and energy storage materials of high energy density systems, CNT-PCs play a crucial role because of their exceptional thermal stability and heat dissipation capabilities. These electronic components generate a lot of heat, which requires thermal energy dissipation through so-called spacer materials to ensure working conditions at a suitable temperature. Due to their light weight and low cost. CNT-PCs are an excellent choice for such applications.

Polymeric composites consist of fillers embedded in a polymer matrix, the first ones will significantly affect the overall (macroscopic) performance of the material. There are many common carbon-based fillers such as single-walled carbon nanotubes (SWCNT), multi-walled carbon nanotubes (MWCNT), carbon nanobuds (CNB), fullerene, and graphene [3–7]. As already stated, the focus will be on CNT-PCs. Since experiments are time-consuming, expensive and cannot even discover mechanism taking place for instance at the interfaces/interphases of composites, they are often complemented through theoretical and computational analysis. Furthermore, numerical simulations have been used to guide the experimental process [8,9]. Typical micro-scale approaches for CNT-PCs include mean-field micro-mechanics methods such as the one proposed by Mori-Tanaka (MT) [10]. Alternatively, statistical methods are employed that describe the micro-structures' performance through the concept of effective probability [11]. Those approaches have been exploited for instance by Zhai et al. [12], Sheng et al. [13] and Mortazavi et al. [14] predicting the thermal conductivity of carbon-based composites. The microstructures of CNT-PCs are stochastic in nature. However, most contributions present a deterministic model neglecting uncertainties, which undoubtfully exist for such materials. Consequently, predicted results deviate from experimental ones [15,16]. In order to better understand internal mechanisms of the

* Corresponding author.
E-mail address: timon.rabczuk@uni-weimar.de (T. Rabczuk).

https://doi.org/10.1016/j.compscitech.2022.109425
Received 21 May 2021; Received in revised form 14 March 2022; Accepted 27 March 2022
Available online 18 April 2022

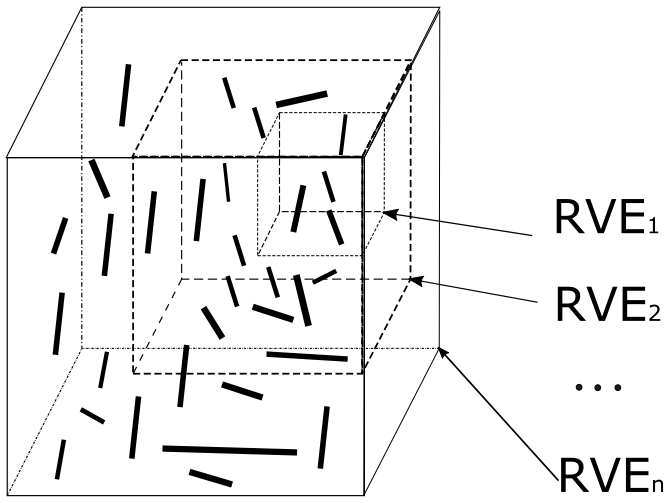


Fig. 1. Sample enlargement method.

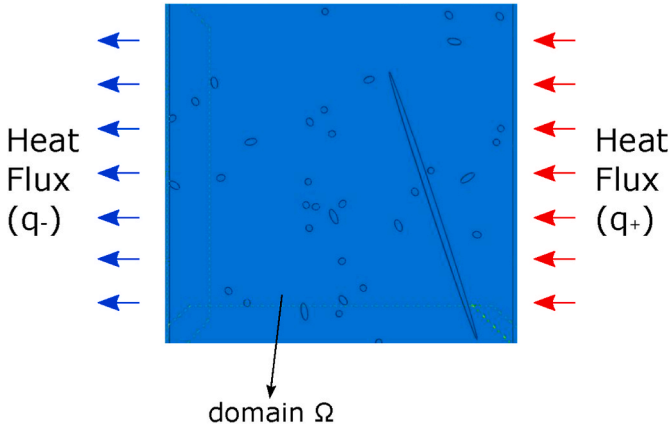


Fig. 2. Heat flux and temperature boundary in RVEs.

material and to determine the key parameters governing the macroscopic response of the composite, uncertainties must be accounted for in the modeling process. Mahmood M.Shokrieh et al. presented a stochastic multi-scale model to analyze the mechanical properties in CNT composites, wherein the fiber's length, the orientation of fiber, agglomeration, curvature, and dispersion of carbon nanotubes are considered as uncertain input parameters [17], see also the contributions by Vu-Bac et al. [18,19]. We previously proposed an uncertainty analysis for stochastic modeling of polymeric nanocomposites (PNCs) [20]. However, the previous work is based on linear and polynomial regression models, which has shortcomings in accuracy and computational efficiency for multi-scale analysis of composite materials with large sample size. Therefore, in this study, we present improved regression models.

As stochastic multiscale models are computational expensive [21], surrogate approaches are commonly exploited that can help propagate uncertain parameters across scales. With the emergence of high-performance computing and artificial intelligence, machine learning has become a popular modeling tool for numerous applications [22]. Machine learning (ML) is commonly used in regression and maps data through specific rules with algorithms to build input and output models. They are particularly useful for nonlinear input-output relationships when sufficient data is available [23]. ML has also been used in the design of new materials and multiscale analysis. Motas et al. [24] for instance took advantage of artificial neural networks (ANN) to predict the multiaxial strain-sensing response of conductive CNT-polymer composites. Qingyuan Rong et al. [25] employed convolutional neural networks (CNNs) for image recognition and data-driven prediction of the effective thermal conductivity of the particle-reinforced composites. Hongwei Guo et al. [26] present a stochastic deep collocation method (DCM) based on neural architecture search (NAS) and transfer learning for heterogeneous porous media. Nguyen-Le et al. [27] presented a combined technique of long short-term memory and hidden Markov model to predict crack patterns. S.Khatir et al. [28] suggested an effective method for crack identification. Roumaissa Zenzen et al. [29] proposed a new modified damage indicator using transmissibility technique combined with ANN to improve Local Frequency Response Ratio (LFCR). Tran-Ngoc et al. [30] proposed a novel machine-learning method based on the global search techniques using vectorized data for damage detection in structures. Wang ei al [31] proposed an algorithm based on deep learning to realize the recognition of different types of rail profiles. Huang et al. [32] adopted a machine learning approach to predict the mechanical properties of carbon nanotube (CNT)-reinforced cementitious composites.

In this manuscript, we present a machine learning based approach accounting for uncertainties to assist stochastic modeling and finally predict the macroscopic thermal conductivity of CNT-PCs based on numerous fine scale features including the Kapitza resistance. This article is organized as follows. In the next section, we describe the stochastic model for CNT-PCs. We introduce our overall framework in section 3, which is based on integrated machine learning and hyperparameters tuning. Subsequently, we discuss the results before we conclude our manuscript in section 4.

2. Stochastic modeling in CNT-PCs

We focus on composites which are manufactured by Chemical Vapor Deposition (CVD). The matrix is an epoxy resin, i.e. cycloaliphatic polyamine (Araldite LY564/Aradur HY2954), the catalyst is zeolite-supported cobalt and iron (HSZ-390HUA) and single-wall CNTs are employed as reinforcement [33–36].

2.1. Homogenization

2.1.1. RVE definition

To determine an 'appropriate' Representative Volume Element (RVE), we employ the sample enlargement method (SEM) [37] as illustrated in Fig. 1. Therefore, we start with a small RVE predicting the effective properties of the homogenized continuum and subsequently increase the RVE sizes. A suitable RVE size is found when the extracted homogenized properties converge to a stationary value, which is checked by

$$\langle R\rangle = \frac{1}{M}\sum_{K=1}^{M} R^{(K)} \tag{1}$$

where M is the number of computations of the ensemble average and $R^{(K)}$ is the performance in the $k-th$ RVE (K = 1,2, …,M). In order to confirm whether the model has reached convergence, the judgment we applied is in Eq (2):

$$\left\|\frac{\langle R^{(K+1)}\rangle - \langle R^{(K)}\rangle}{\langle R^{(K)}\rangle} < Tol = 1\% \right. \tag{2}$$

where $R^{(K)}$ is an apparent performance in $k-th$ RVE, $R^{(K+1)}$ is averaged $K+1$th RVE.

2.1.2. RVE generation algorithm

We take advantage of a previously developed Python script [38] to generate random micro-structures as indicated in Fig. 4. Based on a given probability density functions (PDFs) of the input parameters, the fillers are placed in the RVE ensuring no overlaps. The algorithm also ensures that the RVE satisfies periodicity of the geometry, i.e. the missing part of the fillers on one side of the RVE are placed on the

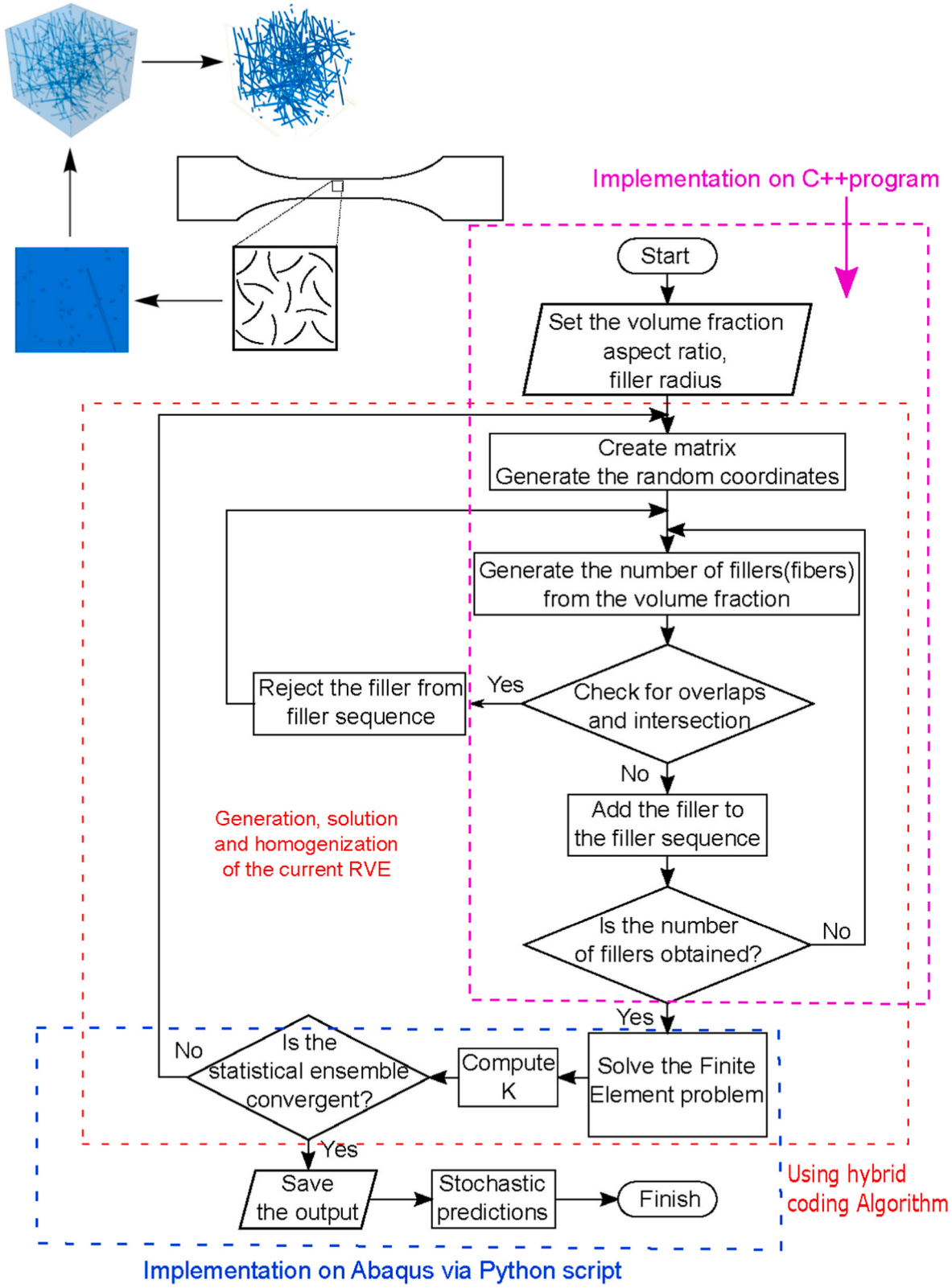


Fig. 3. Flowchart of RVE generation algorithm.

opposite side of the RVE model. Fig. 4 depicts the discretization of the RVE using 4-node tri-linear tetrahedral elements (DC3D4) for both the filler and the matrix. The flowchart of our stochastic modeling approach is summarized in Fig. 3; the dashed box inside this flowchart shows the RVE generation algorithm.

2.1.3. Governing equations

The underlying partial differential equation (PDE) is the quasi-static heat conduction given by

$$\nabla \cdot \boldsymbol{q} - Q = 0 \tag{3}$$

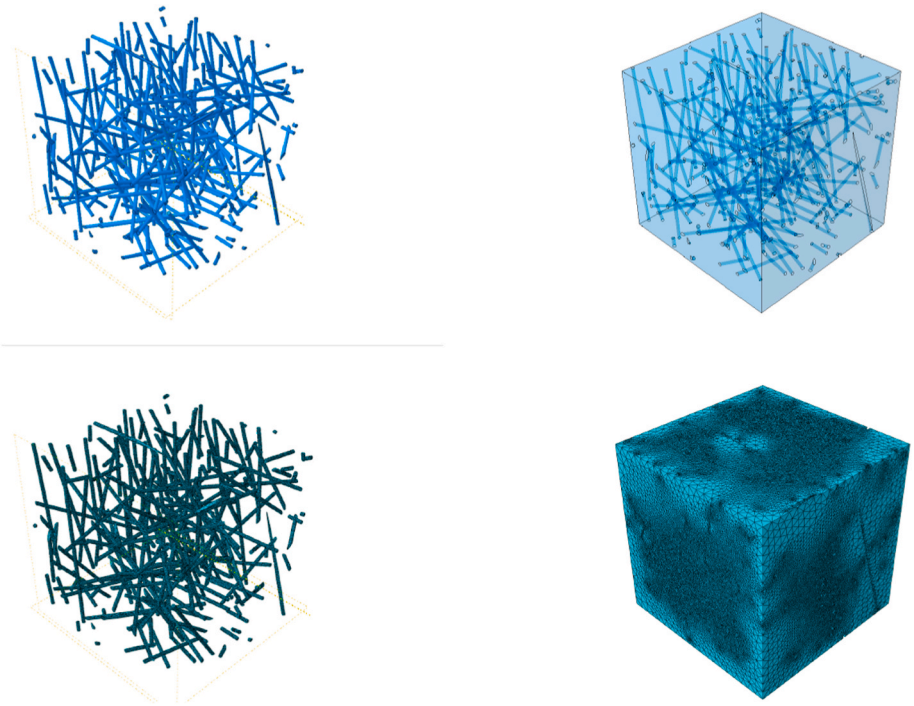
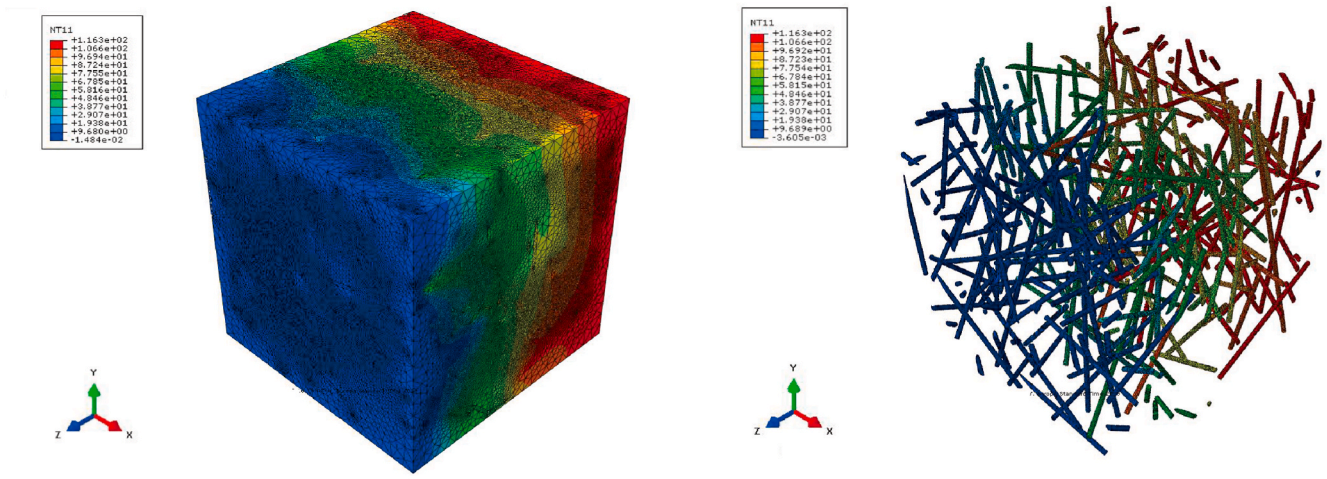


Fig. 4. 3D cubic RVE.



(a) Temperature distribution of composites within RVEs (b) Temperature distribution of inner fibers within RVEs

Fig. 5. The temperature distribution within the RVE.

where $\boldsymbol{q}$ is the heat flux vector and Q is the heat source. Fourier's law relates the heat flux to the temperature θ:

$$\boldsymbol{q} = -\boldsymbol{\kappa}\nabla\theta, \tag{4}$$

$$\text{with} \quad \boldsymbol{\kappa} = \begin{Bmatrix} \kappa_{xx} & 0 & 0 \\ 0 & \kappa_{yy} & 0 \\ 0 & 0 & \kappa_{zz} \end{Bmatrix} \tag{5}$$

We assume that the materials in this study are isotropic, i.e. $\kappa_{xx} = \kappa_{yy} = \kappa_{zz}$, which can be confirmed by applying boundary conditions at different RVE edges. In order to extract the macroscopic (effective) thermal conductivity of the composite, a heat flux ($+q$) is applied on the surface of the RVE cube (along the X-axis) while a negative heat flux ($-q$) is applied on the opposite surface as shown in Fig. 2. This introduces a heat flow from the hot to the cold surface, which in turn creates a temperature gradient $\frac{d\theta}{dx}$. The effective thermal conductivity (ETC) can then be obtained from Fourier law [39] as

$$\boldsymbol{\kappa}^{macro} = \boldsymbol{q} \otimes \frac{d\theta}{d\mathbf{x}} \tag{6}$$

$\frac{d\theta}{dx}$ being the temperature gradient. After averaging the values of all three axes, we can obtain the effective thermal conductivity (ETC) of the RVE.

Substituting equation (3) into equation (4) yields

$$\text{div}(\kappa\nabla\theta) + Q = 0 \qquad in \quad \Omega \tag{7}$$

$$q_n = -\boldsymbol{q}\cdot n = \overline{q} \qquad on \quad \Gamma_q \tag{8}$$

It can be shown that the weak form is given by: Find $\theta \in \mathcal{T}$ with

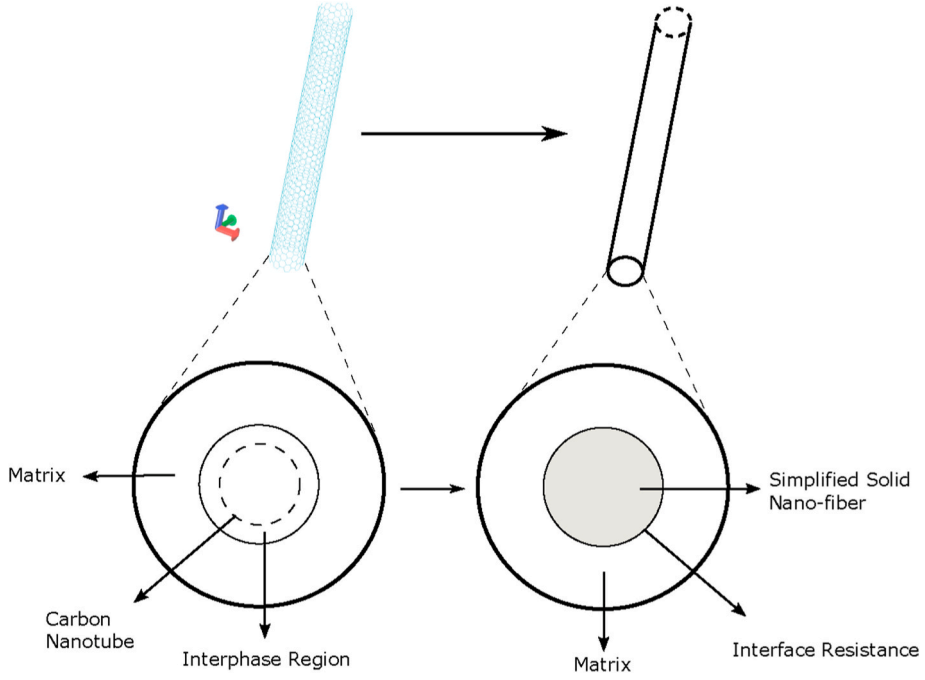


Fig. 6. Illustration of simplified method.

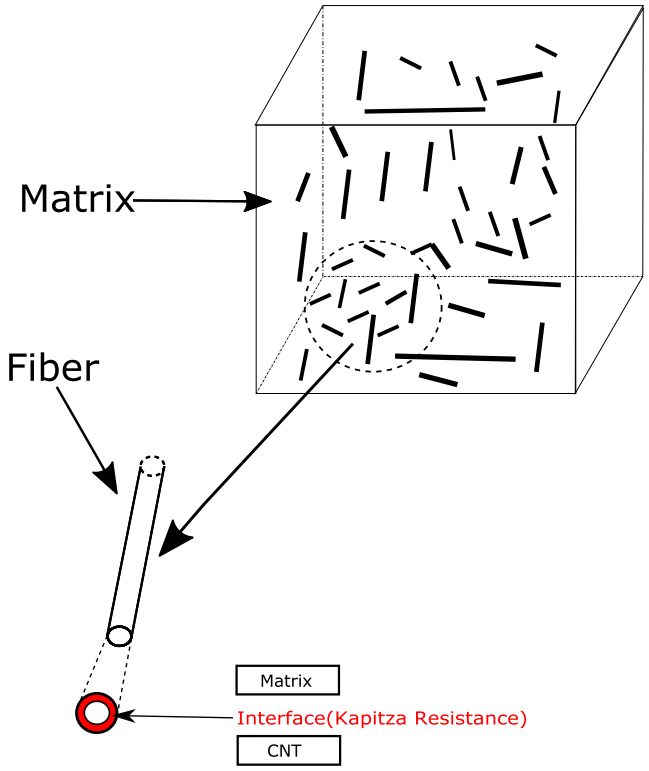


Fig. 7. Kapitza resistance.

admissible test functions $\delta\theta \in \mathcal{V}$ such that

$$\int_{\Omega} \kappa \nabla\theta \cdot \nabla\delta\theta d\Omega = -\int_{\Gamma_q} \delta\theta \overline{q} d\Gamma - \int_{\Gamma_\theta} \delta\theta q_n d\Gamma + \int_{\Omega} \delta\theta Q d\Omega \quad \forall \delta\theta \in \nu \tag{9}$$

where $\overline{q}$ is the imposed flux value on the boundary Γ_q.

We employ the commercial finite element software Abaqus to solve the heat transfer problem. Fig. 5 shows the temperature distribution within an RVE.

2.2. Modeling of uncertainties

In our study, we consider a variety of uncertainties including uncertain material parameters, uncertainties in the micro-structure and uncertainties of the preparation process. We employ a so-called equivalent simplified solid fiber which models the SWCNT including the neighbouring interphase [40] as shown in Fig. 6. It is discretized with Euler Bernoulli beam elements and its effective thermal conductivity is obtained by Fourier law. Due to the geometric configuration of the SWCNTs, the corresponding thermal properties of the equivalent simplified solid fiber were assumed to be transversely isotropic. For simplified solid fiber, the equivalent thermal constants can be estimated by

$$\kappa_{equ} = \frac{A_{CNT}}{A_{equ}} \kappa_{CNT} \tag{10}$$

$$A_{equ} = \pi R_{equ}^2 \tag{11}$$

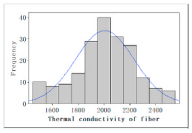
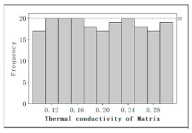
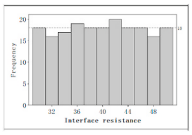
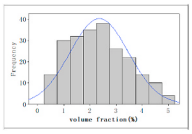
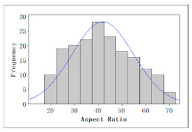
$$A_{CNT} = \pi (R_{CNT}^2 - (R_{CNT} - h_{CNT})^2) \tag{12}$$

where A_{equ} indicates the area of the solid cylinder fiber, A_{CNT} is the cross-sectional area of a hollow SWCNT and R_{equ} is the radius of the solid cylinder; R_{CNT} denotes the radius of the hollow SWCNT and h_{CNT} the distance between the circle center and inner wall of CNT.

We assume a non-deterministic thermal conductivity of the matrix and the equivalent fiber. Also the Kapitza resistance, i.e. interface resistance between the fiber and the matrix is considered as stochastic input parameter. Fig. 7 shows this interface zone between fillers and matrix. Uncertainties concerning the micro-structures are modeled through uncertainties in the aspect ratio while uncertainties of the preparation process are related to the volume fraction. A summary of the non-deterministic input parameters including their PDF can be found in Table 1.

We employ Latin Hypercube Sampling (LHS) [41], which reduces the sampling time by randomly generating multiple input parameters. We consider a model with $m = 5$ input parameters and extract each input

Table 1
Model uncertainties.

Inputs	mean	standard deviation	Type of distribution	Sources
Thermal conductivity of Fiber (X_1)	2006	227.2		Suchismita et al. [43]
Thermal conductivity of Matrix(X_2)	0.20	0.013		A. Moisala et al. [44]
Interface resistance (X_3)	40	0.15		Assumed
Volume fraction (X_4)	0.0235	0.01		M.Shokrieh et al. [17]
Aspect ratio (X_5)	58	1.8		Assumed

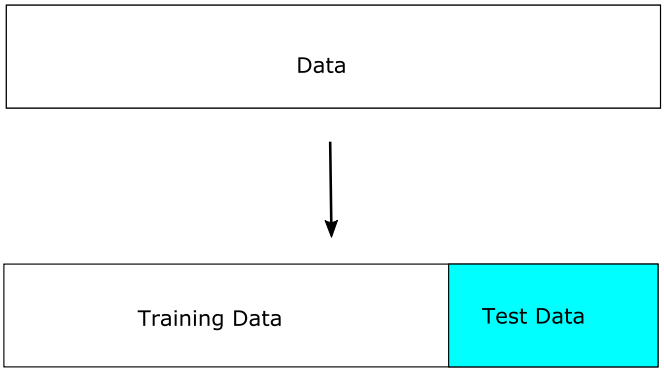


Fig. 8. Holdout cross-validation.

parameter separately and then divide its cumulative probability curve into the same interval of N. After merging all the intervals, a $N \times m$ design matrix is obtained, such that the values are randomly and independently distributed. Afterwards, we obtain the representative generated value by mapping the value of this design matrix to the real physical model [42].

3. Machine learning method

3.1. Dataset preparation

The 'raw data' is obtained directly from our 'model' at different length scales from FE simulations done with ABAQUS. This raw data is divided into two sets: the training set, which accounts for 80% of the overall proportion, and the test set, which accounts for the remaining 20%. Subsequently, we normalize the data to a certain interval [0,1] ensuring that the data of different attributes are at the same scale, which is essential for training the machine learning models since it can significantly improve the training speed. Besides, the data has been checked to ensure the validity [45].

3.2. Performance measurement

The most commonly used approach to describe the predictive ability is Root Mean Square Error (RMSE), which is the difference between the observed and predicted values given by

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(Y_{ri} - Y_{pi})^2} \tag{13}$$

We also compute the Adjusted Coefficient of Determination(R^2_{adj}):

$$R^2 = 1 - \frac{\sum_{j=1}^{N}(Y_{ri} - Y_{pi})^2}{\sum_{j=1}^{N}(Y_{ri} - Y_{mean})^2} \tag{14}$$

$$R^2_{adj} = 1 - \frac{N-1}{N-k_R}(1-R^2) \tag{15}$$

where k_R is the number of predictors in the regression model. Since R^2_{adj} takes the number of training data into account, it accounts for over-fitting and is therefore more reliable than the R^2 value.

3.3. Cross validation

Cross-Validation (CV) is employed in machine learning to build the models and verify its parameters. The cross-validation is based on the repeated application of data. Therefore, data is subdivided and reorganized according to specific rules, in which each new combination is used to train and evaluate the model. Multiple sets of different training and test sets are obtained while a sample in a training set may become a sample in the test set the next time. There are three types of cross-validations: Holdout Cross-Validation, K-fold Cross-Validation, and Leave-One-Out Cross-Validation [46].

3.3.1. Holdout Cross-Validation

Holdout cross-validation divides the original data set into a training set and a testing set as shown in Fig. 8. They can be trained and tested, respectively, and record the final model accuracy as an indicator of the

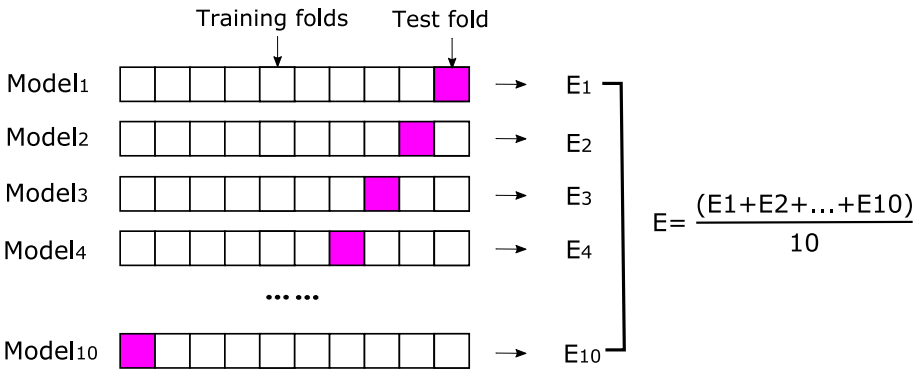


Fig. 9. K-fold cross-validation.

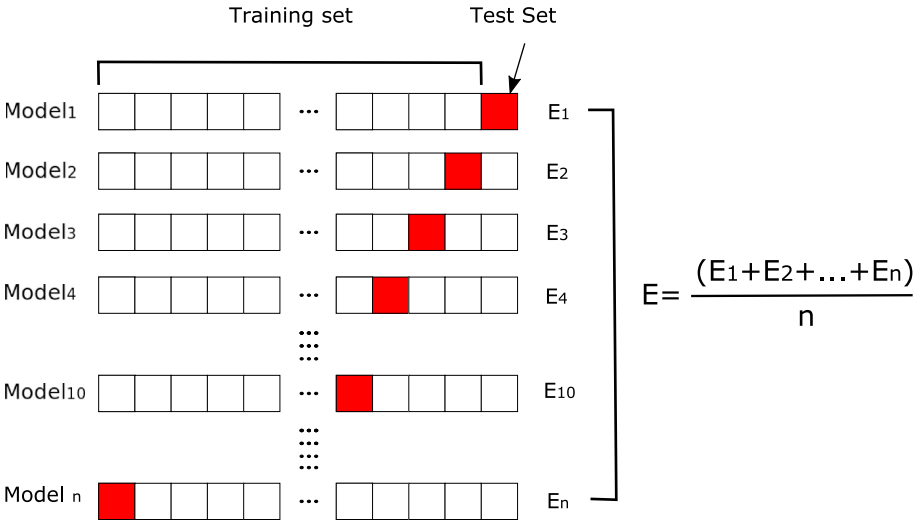


Fig. 10. Leave-one-out cross-validation.

model performance. Holdout verification is an effective method when a large amount of data needs to be verified quickly and easily. When the testing set is large, the training data will be insufficient, so the model is error-prone while a small testing set causes overfitting or underfitting. A typical ratio which yields good results is 80% and 20% [46].

3.3.2. *K-fold cross-validation*

K-fold cross-validation uses a part of the training data for validation. While the validation data is taken from the training data, it does not participate during the training process. It obtains a relatively objective assessment of how well the model matches the data outside the training set. The evaluation of the validation data model is often called cross-validated, also known as circular validation. As indicated by Fig. 9, it divides the original data into K groups (K-Fold) and makes each subset data a validation set, so the remaining $K-1$ group subset data is used as the training set leading to K models. The K models evaluate the validation set results, and the final error RMSE (Root Mean Squared Error) is added and averaged to obtain the cross-validation error. Cross-validation effectively uses limited data, and the evaluation results can be as close as possible to the performance of the model on the test set, which can be used as an index for model optimization [47]. K-fold cross-validation reduces the variance by averaging the results of K different group training, so the model's performance is less sensitive to the division of the data. K-fold cross-validation is applicable when the data set sample is relatively small but the existing data can be fully utilized [48]. The value of K is determined according to the complexity of the data. It commonly ranges from 5 to 10, so in this research, 10-Fold Cross-Validation is employed [49].

3.3.3. *Leave-One-Out Cross-Validation*

Assume that the data set D contains n samples. If $k = n$, a special case of the cross-validation method is obtained: Leave-One-Out Cross-Validation (LOOCV) [50], see Fig. 10. The Leave-One-Out method is not affected by the random sample division method since n samples are uniquely divided into n subsets. The training set of the Leave-One-Out method has only one sample less than the original data set. Thus, the Leave-One-Out method's model is similar to the model trained with D. The Leave-One-Out assessment results are often considered to be more accurate. However, if the data set is large, the computational cost of training n models may be unacceptable. Moreover, the results of the Leave-One-Out method may not always be more accurate than the other evaluation methods [51].

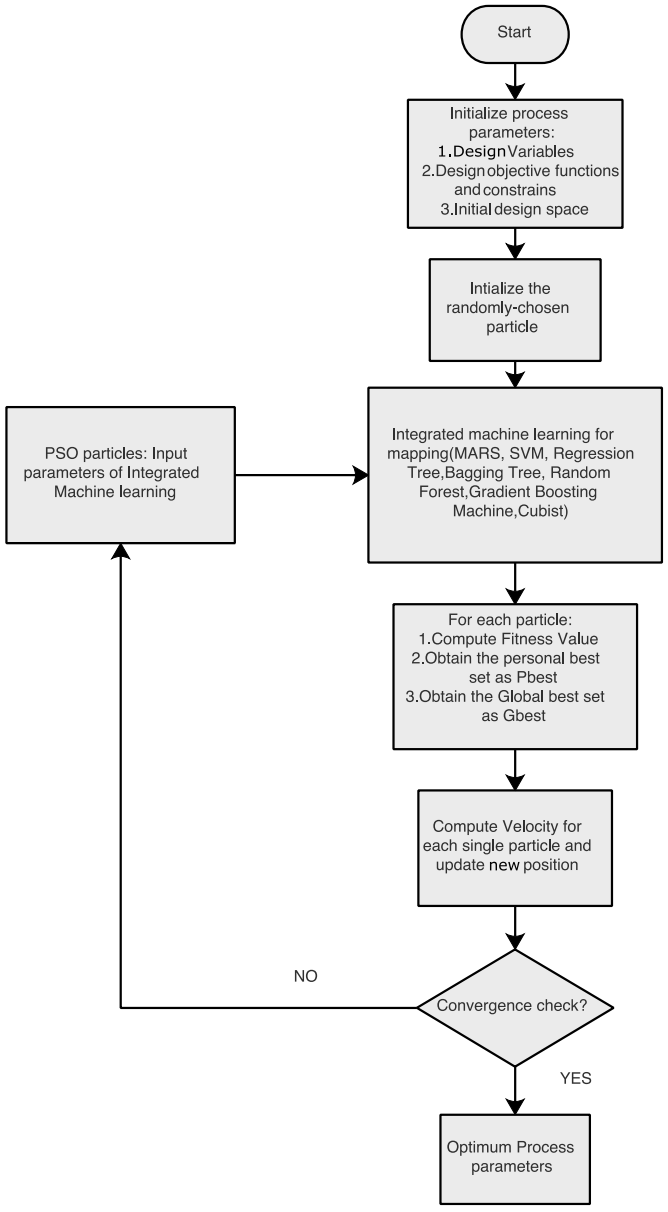


Fig. 11. Flowchart of PSO algorithm in hyper-parameters tuning.

3.4. Hyper-parameter tuning

Hyper-parameters are of great significance for machine learning because the choices of different hyper-parameters significantly impact the prediction performance. Therefore, we take advantage of the Particle Swarm Optimization (PSO) for hyper-parameter tuning. The PSO algorithm is shown as Alg 1, which illustrates how to search for local optima and then iterate to a global optimum. The sum of squared error (SSE) is chosen as PSO's fitness function from 10-fold Cross-Validation (CV); it is minimized continuously during the PSO process. The equation of sum of squared error (SSE), we applied in PSO, is given by

$$F = \frac{1}{N}\sum_{i=1}^{N}(Y_{ri} - Y_{pi})^2 = SSE \tag{16}$$

where F is the fitness function, Y_{ri} and Y_{pi} are the required and predicted i-th output parameters, respectively and N is the number of output parameters. According to our previous research, the swarm size should be chosen as 450; ω, c_1 and c_2 are 1 and (2.0,2.0), respectively [52,53]. Fig. 11 shows a flowchart which can illustrate the entire process during the hyper-parameters tuning.

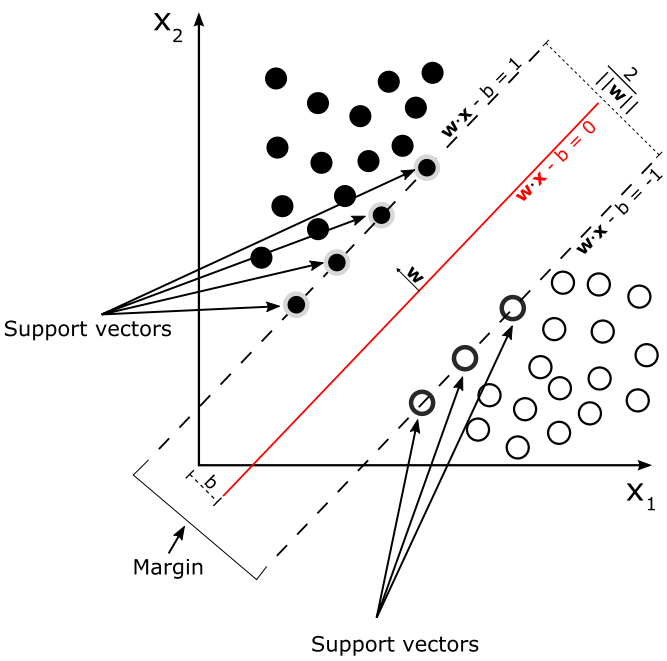


Fig. 12. The illustration of hyper-plane in SVM.

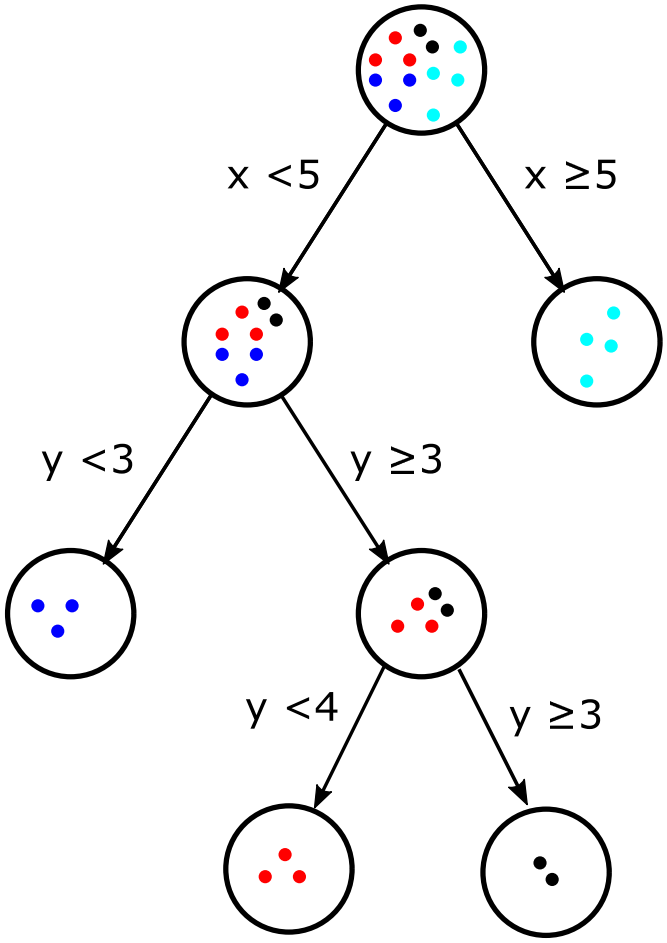


Fig. 13. The structure of Regression tree.

Algorithm 1. Particle Swarm Optimization

Algorithm 1 Particle Swarm Optimization

Require: i,V_i,X_i,N

Ensure: $gBest$

for each particle i **do**

Initialize velocity V_i and position X_i for particle i;

Evaluate particle i and set the $pBest_i$=X_i;

end for

$gBest$=min($pBest_i$);

while Not Stop Running **do**

for i=1 to N **do**

Update the velocity and position of particle i;

Evaluate particle i;

if fit(X_i)<$pBest_i$ **then**

$pBest_i$=X_i;

end if

if $pBest_i$<fit($gBest$) **then**

$gBest$= $pBest_i$;

end if

end for

end while

Print $gBest$;

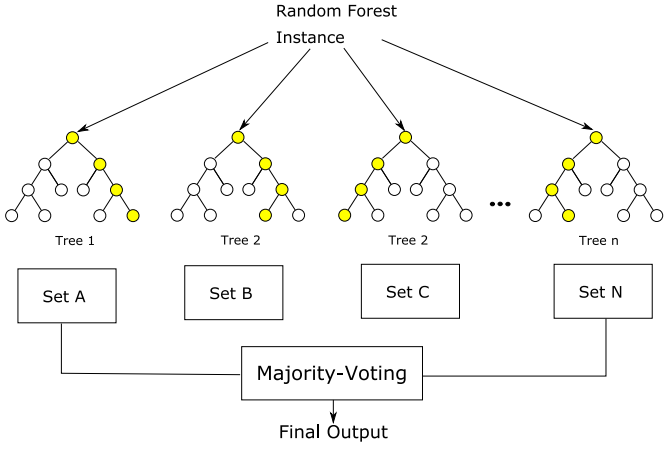


Fig. 14. The architecture of Random Forest.

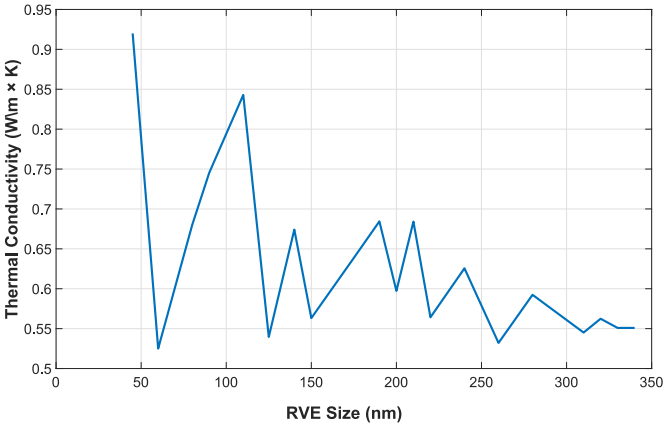


Fig. 15. Thermal conductivity versus RVE size.

3.5. Integrated machine learning

3.5.1. Multivariate Adaptive Regression Splines

Multivariate Adaptive Regression Splines (MARS) is a data analysis method proposed by Jerome Friedman [54]. It is based on spline functions and consists of three steps: forward process, backward pruning process, and model selection. The advantages of MARS include the ability to process large datasets even in high-dimensions and fast computation with superior accuracy. MARS can be seen as a generalization of stepwise linear regression, as it uses the expansion of piecewise linear basis functions of the form $(x-t)_+$ and $(t-x)_+$, where '+' means positive part, i.e.

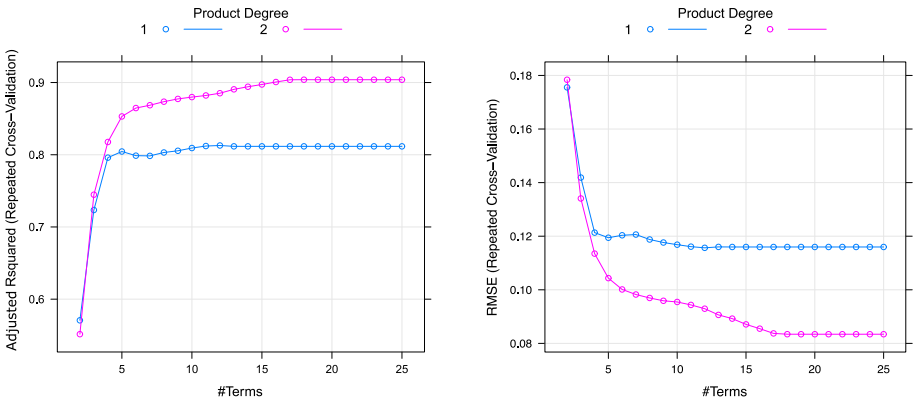
$$(x-t)_+ = \begin{cases} x-t & \text{if } x > t \\ 0 & \text{otherwise} \end{cases} \tag{17}$$

$$(t-x)_+ = \begin{cases} t-x & \text{if } x < t \\ 0 & \text{otherwise} \end{cases} \tag{18}$$

Equations (17) and (18) show a linear spline with a knot at t. In this case, these two functions are called reflected pair; all observations of this input variable are used as knots for each input variable X_j, so the basis function set reads

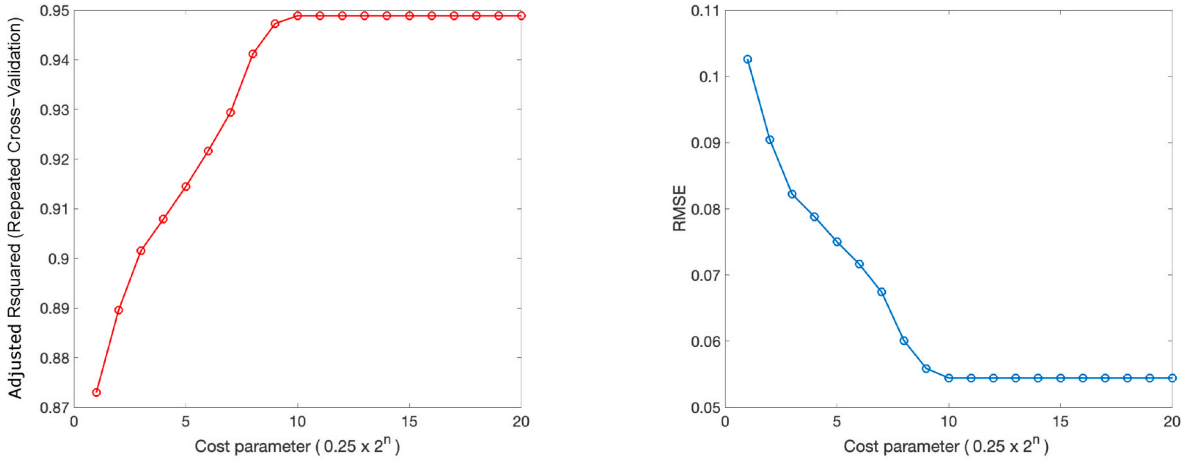
$$C = \left\{(X_j - t)_+, (t - X_j)_+\right\}_{t\in\{x1j,x2j,\ldots,xNj\}\, j=1,2,\ldots p} \tag{19}$$

The strategy for building a model is similar to the forward stepwise linear regression, i.e.



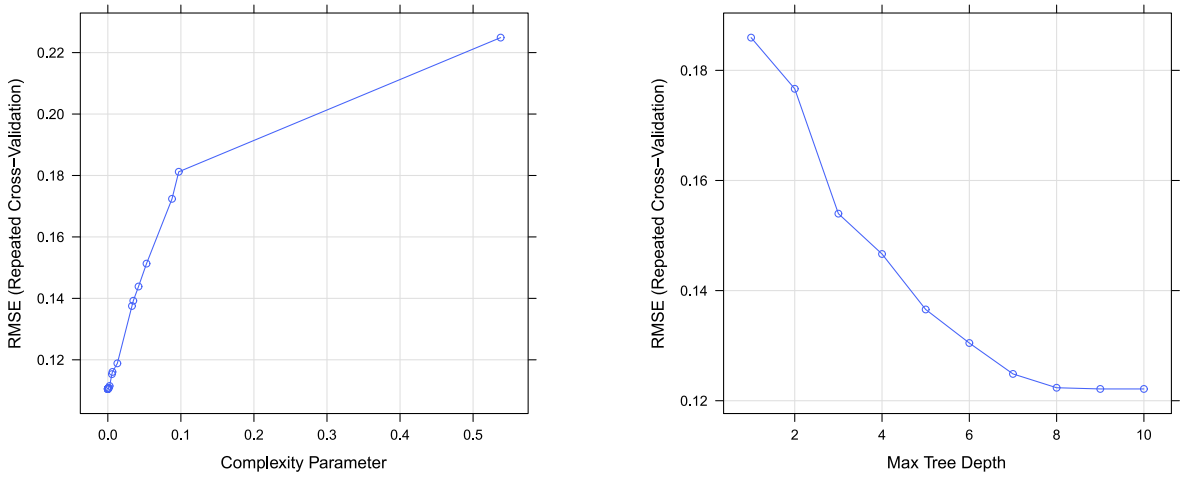
(a) The metric is R^2_{adj} (b) The metric is RMSE

Fig. 16. Hyper-parameter tuning in MARS method with different metrics.



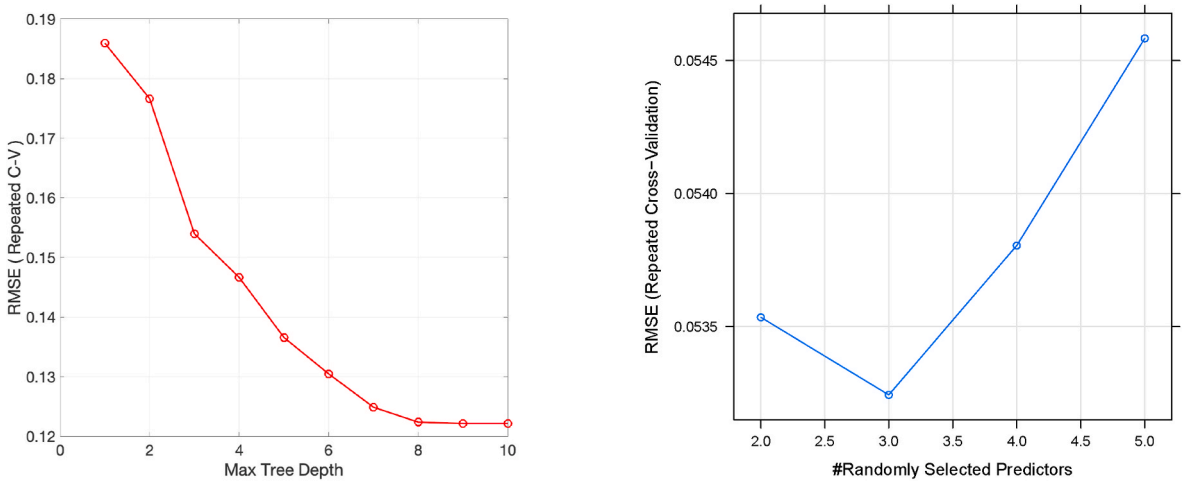
(a) The metric is R^2_{adj} (b) The metric is RMSE

Fig. 17. Hyper-parameter tuning in SVM regression with different metrics.



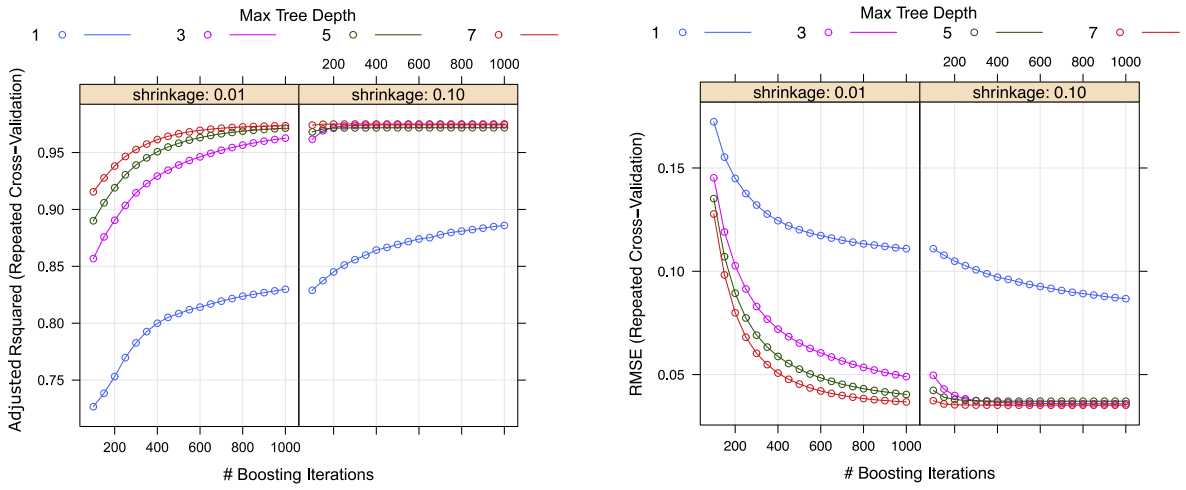
(a) Conplexity Parameter as hyper-parameter (b) Max tree depth as hyper-parameter

Fig. 18. Hyper-parameter tuning in Regression Tree in metric RMSE.



(a) Max tree depth as hyper-parameter (b) The minimum samples for split

Fig. 19. Hyper-parameter tuning in Random Forest.



(a) The metric is R^2_{adj} (b) The metric is RMSE

Fig. 20. Hyper-parameter tuning in GBM method with different metrics.

$$f(x) = \beta_0 + \sum_{m=1}^{M} \beta_m h_m(X) \tag{20}$$

where $h_m(x)$ indicates a basis function in C (or a multiplication of basis functions). Then, we can use the minimum sum of squared residuals through standard linear regression to estimate the coefficient β_m.

3.5.2. Support Vector Machine

Support vector machine (SVM) is a binary classification model based on margin maximization, solving a convex quadratic functional, which is equivalent to minimizing the regularized hinge loss function. SVMs search for a separating hyper-plane that can correctly divide the training data set and obtain the largest geometric interval. In a linearly separable data set, those hyper-planes are unlimited, but the separating hyper-plane with the most considerable geometric interval is unique, see Fig. 12.

The judgment criteria for a sample (x, y) is different from traditional regression problems. In SVM regression, the loss is calculated only when the absolute difference between $f(x)$ and y is greater than ε. This is equivalent to constructing a margin area with a width of 2ε centering on $f(x)$ in space, which the training samples falling into this margin area are regarded as correct predictions, i.e.

$$\arg\min_{\omega,b} \quad \frac{1}{2}\|\|\omega^2 + C\sum_{i=1}^{m} l_\varepsilon(f(x_i) - y_i) \tag{21}$$

where C is a regularization constant, l_ε refers to the ε-insensitive loss function:

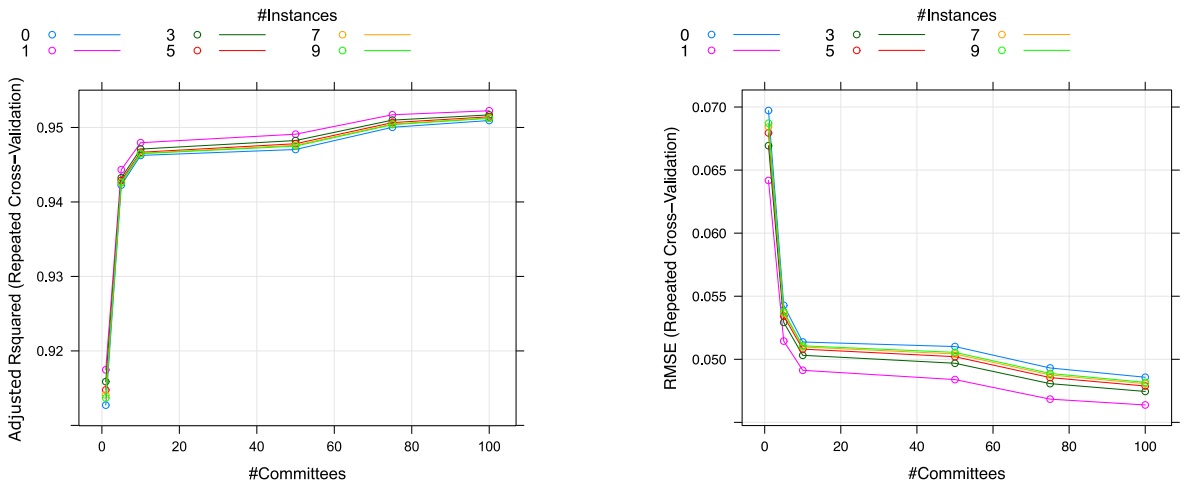
$$l_\varepsilon(z) = \begin{cases} 0, & if|z| \leqslant \varepsilon \\ |z| - \varepsilon, & otherwise \end{cases} \tag{22}$$

The solution of Eq (22) requires the Lagrange operator α_i leading to

$$f(x) = \sum_{i=1}^{m} (\widehat{\alpha}_i - \alpha_i) x_i^T x + b \tag{23}$$

The non-linear data in the original space can be mapped to the data of the approximate linear relationship in the high-dimensional space through the kernel function $\kappa(x, x_i)$ yielding

$$f(x) = \sum_{i=1}^{m} (\widehat{\alpha}_i - \alpha_i) \kappa(x, x_i) + b \tag{24}$$



(a) The metric is R^2_{adj} (b) The metric is RMSE

Fig. 21. Hyper-parameter tuning in Cubist method with different metrics.

Table 2
Hyper-parameters tuning.

ML method	hyper-parameters	Definition	Interval	Optimum Value
Multivariate Adaptive Regression Splines	D	The degree of the features that are added to the model	[0,2]	2
	N_{terms}	The number of retained terms	[2,25]	19
Support Vector Machine	C	Cost parameter	[0.25,131072]	128
	σ	scale in Radial basis function	[0.001,0.5]	0.2866468
Regression Tree	C_p	complexity parameter	[1.862236e-06,5.377315e-01]	1.862236e-06
	M	The maximum depth of tree	[1,10]	9
Bagging Tree	None	None	None	None
Random Forest	Min-sample-split	The minimum samples for split	[2,5]	3
	M	The maximum depth of tree	[1,10]	9
	Max-DT	The maximum numbers of regression tree	[1,1000]	1000
Gradient Boosting Machine	λ	The learning rate	[0.001,0.99]	0.1
	N	The maximum number of regression tree	[100,1000]	950
	D_{ia}	Interaction depth	[1,10]	7
Cubist	C	An optional boosting - like procedure called committees	[1,100]	100
	N	The number of neighbors in KNN	[0,10]	1

There are many commonly used kernel functions, such as the linear kernel, polynomial kernel, Sigmoid kernel, Laplacian kernel and RBF kernel. In our model, the RBF kernel is employed.

3.5.3. Regression tree

The tree-based model establishes a connection between inputs and outputs by dividing the data multiple times. Its structure is illustrated in Fig. 13. This tree-based model simulates the process of human decision making and in this sense differs from other machine learning models. It is based on one or more if-then judgments, named divide-and-conquer strategy, where each judgment corresponds to an if-then judgement. The division of data needs to be homogeneous. The tree-based model consist of the following three steps/ingredients: i) Predictor variables for segmentation and corresponding segmentation points; ii) Depth and complexity of the tree; iii) Prediction equations at the final node.

There are several approaches to construct a regression tree. We use the method from Breiman [55], that starts with a complete data set S and gradually searches for different values of each predictor variable. Then the data is divided into S_1 and S_2 groups by minimizing the following equation:

$$SSE = \sum_{i \in S_1} (y_i - \overline{y_1})^2 + \sum_{i \in S_2} (y_i - \overline{y_2})^2 \tag{25}$$

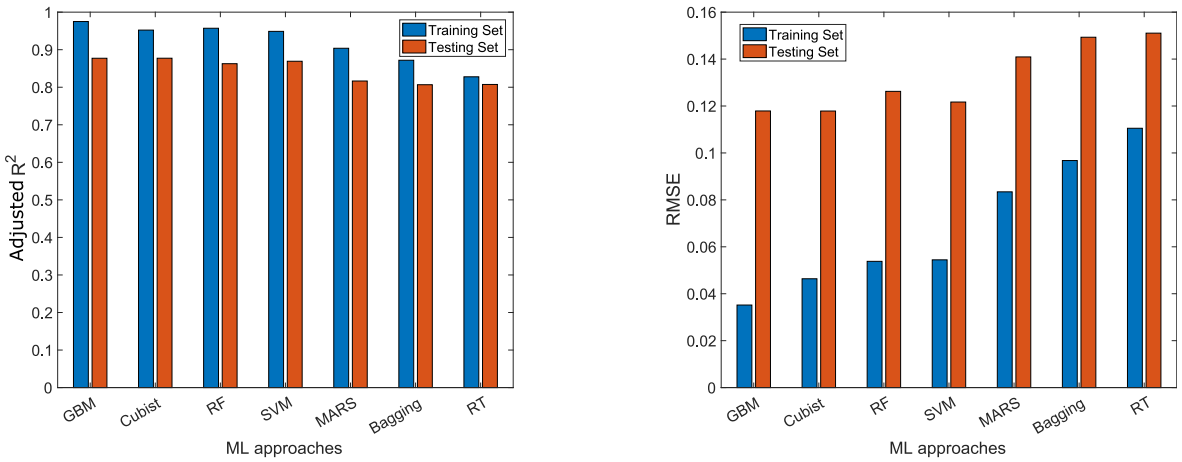
where SSE is the sum of the squared error; $\overline{y_1}$ and $\overline{y_2}$ are the averages of the training set results in the groups S_1 and S_2; y_i denotes the value of the $i-th$ y. The model searches for predictor variables and cuts points in S_1 and S_2, so that the SSE reaches the maximum reduction. Trees need to be pruned to avoid over-fitting. Therefore a penalty is introduce yielding

$$SSE_{cp} = SSE + c_p \times N_{fn} \tag{26}$$

where c_p is a complexity parameter and N_{fn} is the number of final nodes. The model can be tuned by selecting the complexity parameter through trial and error so that the value of RMSE is minimized.

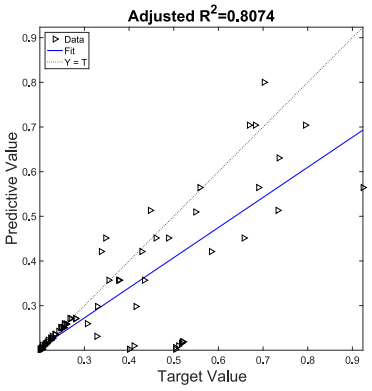
3.5.4. Bagging Tree

Bagging [56] is parallel inheritance learning method and is based on bootstrap sampling. Given a data set of m samples, one sample is randomly taken and put back into the data set later. Therefore, a sample set containing m samples can be established after m random sampling.

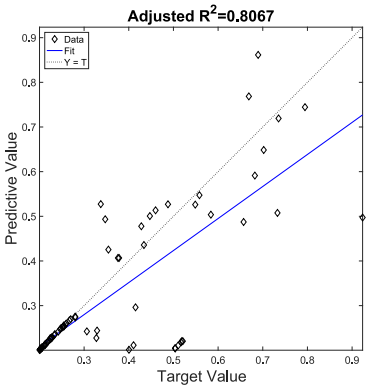


(a) The metric is R^2_{adj} (b) The metric is RMSE

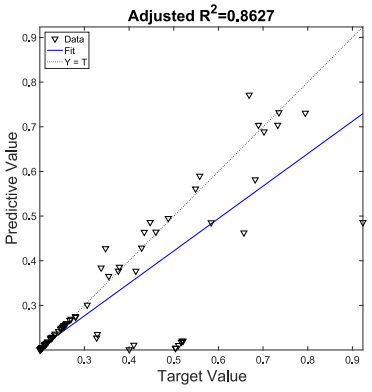
Fig. 22. Predictive Performance in different ML models.



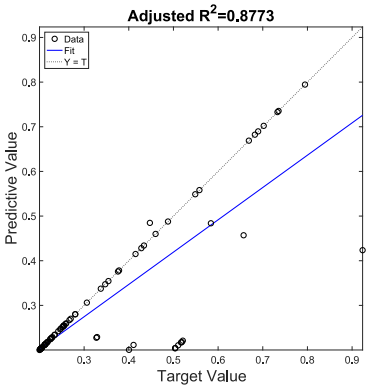
(a) The performance of Regression Tree in testing set.



(b) The performance of Bagging Tree in testing set.

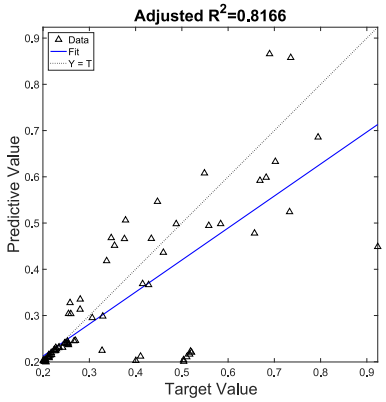


(c) The performance of Random Forest in testing set.

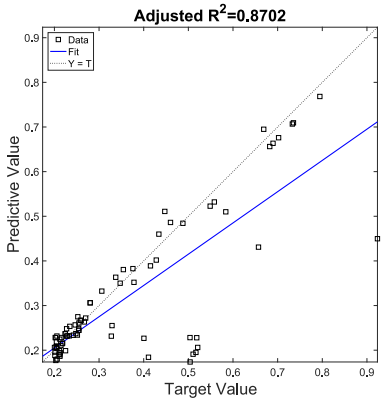


(d) The performance of Gradient Boosting Machine in testing set.

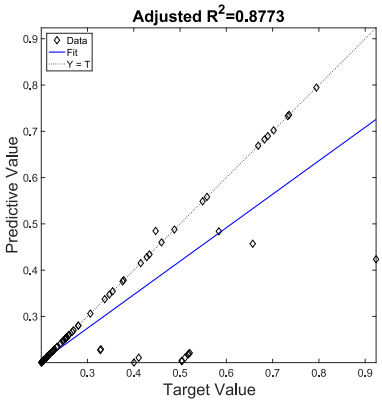
Fig. 23. The performance of tree-based models in testing set.



(a) The performance of MARS in testing set.



(b) The performance of Support Vector Machine in testing set.



(c) The performance of Cubist in testing set.

Fig. 24. The performance rule-based models in testing set.

Table 3
Time complexity in ML models(n is the samples number).

ML method	Time complexity	Computation time
Multivariate Adaptive Regression Splines	$O(n)$	76.973s
Support Vector Machine	$O(m^2 \sim m^3 \times n)$	32.415s
Regression Tree	$O(log_2 n)$	10.606s
Bagging Tree	$O(log_2 n)$	11.609s
Random Forest	$O(M(mn\ logn))$	115.943s
Gradient Boosting Machine	$O(mnlogn)$	198.08s
Cubist	$O(n^3)$	877.327s

One can obtain T sample sets containing m training samples. Subsequently, each component learner is trained based on each sample set. Afterwards, these component learners are assembled. An associated algorithm is presented as Alg 2.

Algorithm 2. Bagging Tree Algorithm

Algorithm 2 Bagging Tree Algorithm

Require: Training set $D = (x_1, y_1), (x_2, y_2), ..., (x_m, y_m)$; Component Learning algorithm L; Training epoch T;

Ensure: $H(x) = argmax \sum_{t=1}^{T} h_t(x) = y$;

1: **For** t=1,2,...,T **do**

2: $h_t = L(D, D_{bs})$

3: **end for**

4: **return** $H(x)$;

The bagging model has the following advantages over other models: i) It reduces the variance of the prediction effectively through the model aggregation process. ii) It can provide an inherent prediction performance estimate, which can be very similar to cross-validation or test set estimation.

3.5.5. Random forest

Random forest is a variation of the bagging method proposed by Breiman [57]. The architecture of this tree-based model is illustrated in Fig. 14. Random attributes are introduced in the training process assuming that the attribute set of the current node consists of d attributes. For each node of the primary decision tree in a random forest, a subset of attributes containing k is randomly selected from the node's attribute set. Then an optimal attribute is chosen from the sub-set for partitioning. The random forest algorithm can be found in Alg 3.

Algorithm 3. Random Forest Algorithm

Algorithm 3 Random Forest Algorithm

Require: The number of predictors k; The number of trees t; The number of split points s;

Ensure: Predictive value $Y(x)$;

```
Choose the number of models m
For t=1,2,...,m do
  Generate a bootstrap sample from the original data
  Train a tree model on this sample
    For s=1,2,...,m do
      Randomly extract k < d attributes as predictors
      Choose the optimal variable among k attributes
    end for
  Tree model rule termination conditions take effect
end for
return Y(x);
```

The random forest approach is easy to implement, computationally inexpensive and shows powerful performance in many tasks. Compared with the bagging method, the random forest approach adds only some self-attribute disturbances based on the sample disturbance to increase component learners' diversity. It has the following features: i) The random forest approach does not suffer from overfitting due to the large number of trees. ii) It resists noise in the response variables. iii) It is computationally efficient – compared to most other models. iiii) The influence of the tuning parameters on the prediction results is minor.

3.5.6. Gradient boosting machine

The Boosting method combines a series of weak learners to finally obtain a strong learner. Therefore, a component learner is trained from the initial training set, and then the training samples are adjusted according to the performance of this component learner. In this adjustment, more attention is paid to the analysis of the 'wrong' samples, so the adjusted samples are trained as the next learner. Once a convergence condition is satisfied, all the component learners are combined. The pseudo-code of this algorithm is summarized in 4. The basic principle is to find a cumulative model minimizing a given loss function under a determined decision tree model. This algorithm uses a prediction value that is the best for a predicted variable as initialization and calculates the residual gradient.

The boosting method has a certain similarity with the random forest approach and integrates the tree-based model through specific rules. However, there are several differences, i.e. all trees in the random forest are built independently, and each tree contributes to the final result. In the boosting rule, subsequent trees depend on the previous and the final result also introduces weights to measure the contribution.

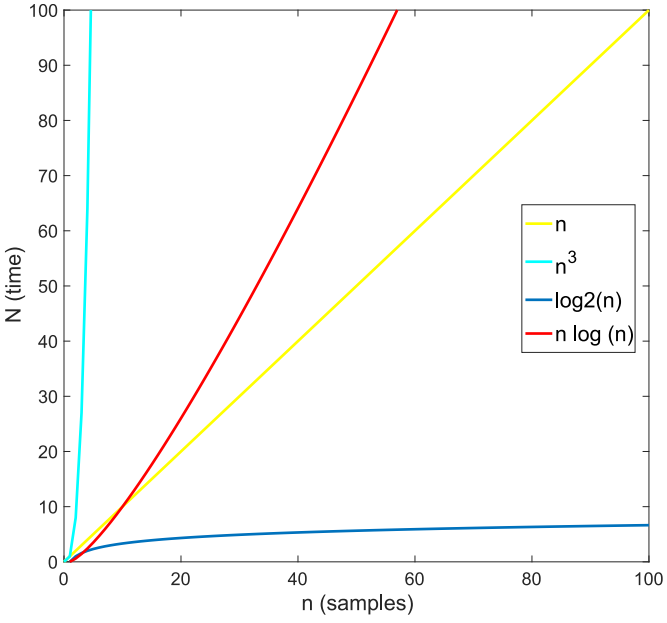


Fig. 25. The complexity in ML models.

Algorithm 4. Gradient Boosting Machine

Algorithm 4 Gradient Boosting Machine

Require:

The Depth of trees D;

The number of iterate K;

Ensure: Predictive value $Y(x)$;

Calculate the mean of the response variable as the initial predicted value of each sample

For k=1,2,...,K **do**

Calculate the residual between observed value and predictive value

Use residuals as response variables in order to fit a regression tree with depth D

Use the previous steps to get the regression tree to predict each sample

Update the predicted value of each sample and add the predicted value obtained in the previous step

end for

return $Y(x)$;

3.5.7. Cubist

Cubist is a rule-based hybrid model that integrates several predictive models [49]. Similar to the boost method, it defines a group-a committee; the predicted values generated by the model can apply the KNN method to adjust the training set. In Cubist, the prediction values are obtained through linear combination, i.e.

$$\widehat{y}_{par} = a \times \widehat{y}_{(k)} + (1-a) \times \widehat{y}_{(p)} \tag{27}$$

where $\widehat{y}_{(k)}$ is the predictive value in the current model and $\widehat{y}_{(p)}$ denotes the predictive value in the main model. The covariance generated by two sets of models are introduced into the smoothing process. According to the linear relationship between the covariance and the residual, we can define the smoothing coefficient as

$$a = \frac{Var[e_{(p)}] - Cov[e_{(k)}, e_{(p)}]}{Var[e_{(p)} - e_{(k)}]} \tag{28}$$

The magnitude of the RMSE value has a specific relationship with the weight of the model. The smaller the RMSE, the larger the smoothed weight. Cubist aggregates a series of linear models on each node into a single smooth model. The committee of this model also generates a series of rule-based models to build on the above basis:

$$y^{*}_{(m)} = y - (\widehat{y}_{(m-1)} - y) \tag{29}$$

The K-nearest neighbor method (KNN) can be applied to adjust the predictive value. When predicting a new sample, K neighbor values need to be found similar to the sample from the training set:

$$\frac{1}{K}\sum\nolimits_{l=1}^{K} \omega_l [t_l + (\widehat{y} - \widehat{t}_l)] \tag{30}$$

where t_l is an observed value of neighboring results, $\widehat{t}_l$ refers to a predictive value in those neighbors and ω_l is the weight. The initial value of the weight is given by

$$\omega_l = \frac{1}{D_l + 0.5} \tag{31}$$

D_l denoting the distance from the nearest neighbor to the predicted sample. The implementation of Cubist model is more complex compared to the previous approaches. Its advantage is the computational efficiency and high accuracy.

4. Results and discussion

4.1. Hyper-parameters tuning

Fig. 15 shows the macroscopic thermal conductivity for different RVE sizes. Subsequently, we present results for an RVE size of 340 nm. Let us start with the hyper-parameters tuning with PSO. Fig. 16 shows the results for MARS. The optimal value occurs for *Degree* = 2. After the 19th term, the RMSE value is stable. Consequently, we will use *terms* = 19 and *Degree* = 2. Fig. 17 illustrates the outcome of the SVM Regression. Based on the RMSE, a C value of $0.25 \times 2^{10} = 128$ seems reasonable, while the scale factor σ was chosen as 0.2866468. The outcome for the different tree-based approaches presented from sections 3.5.3 - 3.5.6 is presented in Figs. 18–20. Hyper-parameter tuning in Cubist is shown in Fig. 21, where the interaction depth is 7, committees are 100, and the number of neighbors is 1. All data with the various hyper-parameters are summarized in the following Table 2

4.2. Integrated machine learning

Seven different machine learning methods are employed to predict the thermal conductivity of CNT-PCs. The R^2_{adj} and RMSE are used as performance indicators. All simulations are done on a 2.8 GHz quad-core Intel Core i7 computer. The same 10-fold CV is applied in every ML model.

Fig. 22 compares the R^2_{adj} and RMSE values for all seven machine learning models. In this Fig Fig. 22 a), we can observe that in the part of the training set, the R^2_{adj} of all models is higher than 0.8, which means the fit of the training process to the data is higher than 80%. For these machine learning models, such data can be seen initially as no underfitting. In terms of test sets that are more significant than the training set, are higher than 0.8. This can be regarded as no over-fitting in our trained models. Specifically, once the R^2_{adj} is under 0.8 or a huge gap between training and testing in R^2_{adj}, the ML model is over-fitting. Hence, those data indicate that our model is well-trained and suitable for our prediction in thermal conductivity.

Let us analyze first the test set performance of the tree-based model. As shown in the Fig. 23, the predictive ability of a single decision tree is mediocre, i.e. 0.8074 and 0.8068, respectively. However, the tree-based integration strategy can be significantly improved by integrating the weak learner (decision tree), which is 0.8627 in the random forest and

0.8773 in gradient boosting machine. We can also see from Fig. 23c) and d) that the strategy of integrating weak learners such as RF and GBM greatly improves the prediction accuracy. Compared with RF, GBM introduces weights during the training process in the integration strategy.

As mentioned before, MARS is a non-parametric regression technology, which can be seen as an extension of the linear model of nonlinear automatic modeling, though it is still a linear model. The linear model has certain shortcomings for the prediction of nonlinear systems. Fig. 24 a) shows that the data of MARS in the test set is only 0.8166. Subsequently, SVM, as an outstanding statistical learning model, is used in most regression problems. As can be seen in Fig. 24 b), the prediction performance of 0.8702 is the most accurate value. The selection of a reasonable kernel function can transform the nonlinear problem into a high-dimensional linear problem, thereby effectively reducing the computational complexity and enhancing the fitting ability.

Finally, Cubist is a rule-based model that is an extension of the tree-based model. The tree is reduced to a set of rules, which initially are paths from the top of the tree to the bottom. Rules are eliminated via pruning and/or combined for simplification. The Cubist model can also use a boosting–like scheme called committees where Iterative model trees are created in sequence. As the most complex integration strategy – illustrated in Fig. 24 c), Cubist has the best predictive ability (0.8773) among all ML models.

It can be observed from the Figs. 23 and 24, most of the predicted values are distributed in the area below the $y = x$ line. This shows that the trained models based on our physical model are more conservative in the prediction of the actual data, i.e., the predicted value is lower than the actual value.

Let us introduce a metric - time complexity, which quantifies the amount of time taken by an algorithm to run as a function of the inputs' length. We summarize all the results in Table 3, where the time complexity and computation time are presented, respectively. Cubist has the largest complexity - $O(n^3)$ and thus yields the longest computational cost of 877.327s in model training. Fig. 25 presents the computational cost versus the sample size for different models; RF and GBM both belong to line $nlog(n)$. The complexity of MARS and SVM is $O(n)$ represented by the line n in the figure. The last set of complexity, namely Regression Tree and Bagging Tree, is represented by the line $log_2(n)$, which is the least sensitive to all complexity sample increments. It means that RT and Bagging are the computationally cheapest approaches.

5. Conclusions

We presented a stochastic multi-scale approach to predict the thermal conductivity of CNT-PCs exploiting integrated machine learning (ML). Various ML models including Multivariate Adaptive Regression Splines (MARS), Support Vector Machine (SVM), Regression Tree (RT), Bagging Tree (Bag), Random Forest (RF), Gradient Boosting Machine (GBM) and Cubist are tested considering five stochastic input parameters; the output is the macroscopic thermal conductivity. We took advantage of PSO to optimize the parameters and 10-fold CV to improve the prediction performance. Through stochastic modeling, totally 500 datasets were generated for integrated machine learning, including 400 datasets for training and 100 datasets are used as test sets. The predictive performance of these models is quantified through the RMSE and R^2_{adj}. Furthermore, we measured the CPU time for each model. The conclusions are summarized as follows:

1. All integrated ML models are suitable to predict the macroscopic thermal conductivity of the composite. The best performance (in terms of RMSE and R^2_{adj}) is obtained by Cubist followed by GBM; the most efficient approaches are RF and SVM.
2. The application of PSO in hyper-parameters tuning can quickly locate the best hyper-parameters and significantly improve the prediction performance. Also, applying 10-fold CV in hyper-parameter tuning can effectively reduce over-fitting.
3. Comparing different models in terms of time complexity, RF and SVM also have good accuracy but less computational cost, which are more recommended when the amount of data is large.

The results show this stochastic integrated machine learning approach accounting for uncertainty is applicable to predict the thermal conductivity of CNT-PCs accurately, which plays an important role in the design of engineering components.

Author statement

Bokai Liu: Formal analysis, Validation, Visualization, Methodology.
Vu-Bac Nam: Methodology.
Xiaoying Zhuang: Methodology.
Xiaolong Fu: Methodology.
Timon Rabczuk: Methodology, Conceptualization, Supervision.

Declaration of competing interest

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

Acknowledgment

We gratefully acknowledge the support of the China Scholarship Council (CSC).

References

[1] R.H. Baughman, A.A. Zakhidov, W.A. De Heer, Carbon nanotubes–the route toward applications, Science 297 (2002) 787–792.
[2] K.M. Liew, Z. Pan, L.-W. Zhang, The recent progress of functionally graded cnt reinforced composites and structures, Sci. China Phys. Mech. Astron. 63 (2020) 1–17.
[3] E.T. Thostenson, Z. Ren, T.-W. Chou, Advances in the science and technology of carbon nanotubes and their composites: a review, Compos. Sci. Technol. 61 (2001) 1899–1912.
[4] B. Mortazavi, S. Ahzi, Thermal conductivity and tensile response of defective graphene: a molecular dynamics study, Carbon 63 (2013) 460–470.
[5] B. Mortazavi, Ultra high stiffness and thermal conductivity of graphene like c3n, Carbon 118 (2017) 25–34.
[6] B. Mortazavi, H. Yang, F. Mohebbi, G. Cuniberti, T. Rabczuk, Graphene or h-bn paraffin composite structures for the thermal management of li-ion batteries: a multiscale investigation, Appl. Energy 202 (2017) 323–334.
[7] B. Mortazavi, Z. Fan, L.F.C. Pereira, A. Harju, T. Rabczuk, Amorphized graphene: a stiff material with low thermal conductivity, Carbon 103 (2016) 318–326.
[8] N. Vu-Bac, R. Rafiee, X. Zhuang, T. Lahmer, T. Rabczuk, Uncertainty quantification for multiscale modeling of polymer nanocomposites with correlated parameters, Compos. B Eng. 68 (2015) 446–464.
[9] N. Vu-Bac, T. Lahmer, Y. Zhang, X. Zhuang, T. Rabczuk, Stochastic predictions of interfacial characteristic of polymeric nanocomposites (pncs), Compos. B Eng. 59 (2014) 80–95.
[10] T. Mori, K. Tanaka, Average stress in matrix and average elastic energy of materials with misfitting inclusions, Acta Metall. 21 (1973) 571–574.
[11] A.K. Sen, S. Torquato, Effective conductivity of anisotropic two-phase composite media, Phys. Rev. B 39 (1989) 4504.
[12] S. Zhai, P. Zhang, Y. Xian, P. Yuan, D. Yang, Modelling and analysis of effective thermal conductivity for polymer composites with sheet-like nanoparticles, J. Mater. Sci. 54 (2019) 356–369.
[13] N. Sheng, M.C. Boyce, D.M. Parks, G. Rutledge, J. Abes, R. Cohen, Multiscale micromechanical modeling of polymer/clay nanocomposites and the effective clay particle, Polymer 45 (2004) 487–506.
[14] B. Mortazavi, M. Baniassadi, J. Bardon, S. Ahzi, Modeling of two-phase random composite materials by finite element, mori–tanaka and strong contrast methods, Compos. B Eng. 45 (2013) 1117–1125.
[15] M.A. Maghsoudlou, R.B. Isfahani, S. Saber-Samandari, M. Sadighi, Effect of interphase, curvature and agglomeration of swcnts on mechanical properties of polymer-based nanocomposites: experimental and numerical investigations, Compos. B Eng. 175 (2019), 107119.
[16] S.I. Yengejeh, S.A. Kazemi, A. Öchsner, Carbon nanotubes as reinforcement in composites: a review of the analytical, numerical and experimental approaches, Comput. Mater. Sci. 136 (2017) 85–101.

[17] M.M. Shokrieh, R. Rafiee, Stochastic multi-scale modeling of cnt/polymer composites, Comput. Mater. Sci. 50 (2010) 437–446.
[18] N. Vu-Bac, T. Lahmer, H. Keitel, J. Zhao, X. Zhuang, T. Rabczuk, Stochastic predictions of bulk properties of amorphous polyethylene based on molecular dynamics simulations, Mech. Mater. 68 (2014) 70–84.
[19] N. Vu-Bac, M. Silani, T. Lahmer, X. Zhuang, T. Rabczuk, A unified framework for stochastic predictions of mechanical properties of polymeric nanocomposites, Comput. Mater. Sci. 96 (2015) 520–535. Special Issue Polymeric Composites.
[20] B. Liu, N. Vu-Bac, X. Zhuang, T. Rabczuk, Stochastic multiscale modeling of heat conductivity of polymeric clay nanocomposites, Mech. Mater. 142 (2020), 103280.
[21] M. Pinsky, S. Karlin, An Introduction to Stochastic Modeling, Academic press, 2010.
[22] E. Alpaydin, Introduction to Machine Learning, MIT press, 2020.
[23] C.M. Bishop, Pattern Recognition and Machine Learning, springer, 2006.
[24] M. Matos, S. Pinho, V. Tagarielli, Application of machine learning to predict the multiaxial strain-sensing response of cnt-polymer composites, Carbon 146 (2019).
[25] Q. Rong, H. Wei, X. Huang, H. Bao, Predicting the effective thermal conductivity of composites from cross sections images using deep learning methods, Compos. Sci. Technol. 184 (2019), 107861.
[26] H. Guo, X. Zhuang, P. Chen, N. Alajlan, T. Rabczuk, Stochastic deep collocation method based on neural architecture search and transfer learning for heterogeneous porous media, Eng. Comput. (2022) 1–26.
[27] D.H. Nguyen-Le, Q. Tao, V.-H. Nguyen, M. Abdel-Wahab, H. Nguyen-Xuan, A data-driven approach based on long short-term memory and hidden markov model for crack propagation prediction, Eng. Fract. Mech. 235 (2020), 107085.
[28] S. Khatir, D. Boutchicha, C. Le Thanh, H. Tran-Ngoc, T. Nguyen, M. Abdel-Wahab, Improved ann technique combined with jaya algorithm for crack identification in plates using xiga and experimental analysis, Theor. Appl. Fract. Mech. 107 (2020), 102554.
[29] R. Zenzen, S. Khatir, I. Belaidi, C. Le Thanh, M.A. Wahab, A modified transmissibility indicator and artificial neural network for damage identification and quantification in laminated composite structures, Compos. Struct. 248 (2020), 112497.
[30] H. Tran-Ngoc, S. Khatir, T. Le-Xuan, G. De Roeck, T. Bui-Tien, M.A. Wahab, A novel machine-learning based on the global search techniques using vectorized data for damage detection in structures, Int. J. Eng. Sci. 157 (2020), 103376.
[31] S. Wang, H. Wang, Y. Zhou, J. Liu, P. Dai, X. Du, M.A. Wahab, Automatic laser profile recognition and fast tracking for structured light measurement using deep learning and template matching, Measurement 169 (2021), 108362.
[32] J. Huang, J. Liew, K. Liew, Data-driven machine learning approach for exploring and assessing mechanical properties of carbon nanotube-reinforced cement composites, Compos. Struct. 267 (2021), 113917.
[33] S. Maruyama, R. Kojima, Y. Miyauchi, S. Chiashi, M. Kohno, Low-temperature synthesis of high-purity single-walled carbon nanotubes from alcohol, Chem. Phys. Lett. 360 (2002) 229–234.
[34] S. Harish, K. Ishikawa, E. Einarsson, S. Aikawa, S. Chiashi, J. Shiomi, S. Maruyama, Enhanced thermal conductivity of ethylene glycol with single-walled carbon nanotube inclusions, Int. J. Heat Mass Tran. 55 (2012) 3885–3890.
[35] A. Vavouliotis, E. Fiamegou, P. Karapappas, G. Psarras, V. Kostopoulos, Dc and ac conductivity in epoxy resin/multiwall carbon nanotubes percolative system, Polym. Compos. 31 (2010) 1874–1880.
[36] E. Fiamegkou, N. Athanasopoulos, V. Kostopoulos, Prediction of the effective thermal conductivity of carbon nanotube-reinforced polymer systems, Polym. Compos. 35 (2014) 1997–2009.
[37] M. Silani, H. Talebi, S. Ziaei-Rad, P. Kerfriden, S.P. Bordas, T. Rabczuk, Stochastic modelling of clay/epoxy nanocomposites, Compos. Struct. 118 (2014) 241–249.
[38] B. He, B. Mortazavi, X. Zhuang, T. Rabczuk, Modeling kapitza resistance of two-phase composite material, Compos. Struct. 152 (2016) 939–946.
[39] B. Mortazavi, J. Bardon, S. Ahzi, Interphase effect on the elastic and thermal conductivity response of polymer nanocomposite materials: 3d finite element study, Comput. Mater. Sci. 69 (2013) 100–106.
[40] M.M. Shokrieh, R. Rafiee, Prediction of mechanical properties of an embedded carbon nanotube in polymer matrix based on developing an equivalent long fiber, Mech. Res. Commun. 37 (2010) 235–240.
[41] R.L. Iman, W. Conover, Small sample sensitivity analysis techniques for computer models. with an application to risk assessment, Commun. Stat. Theor. Methods 9 (1980) 1749–1842.
[42] D. Novák, B. Teplý, Z. Keršner, The role of Latin hypercube sampling method in reliability engineering, in: Proc. Of ICOSSAR, volume vol. 97, pp. 403–409.
[43] S. Ghosh, W. Bao, D.L. Nika, S. Subrina, E.P. Pokatilov, C.N. Lau, A.A. Balandin, Dimensional crossover of thermal transport in few-layer graphene, Nat. Mater. 9 (2010) 555.
[44] A. Moisala, Q. Li, I. Kinloch, A. Windle, Thermal and electrical conductivity of single-and multi-walled carbon nanotube-epoxy composites, Compos. Sci. Technol. 66 (2006) 1285–1288.
[45] S. Xu, J. Wang, W. Shou, T. Ngo, A.-M. Sadick, X. Wang, Computer vision techniques in construction: a critical review, Arch. Comput. Methods Eng. 28 (2021) 3383–3397.
[46] R. Kohavi, et al., A study of cross-validation and bootstrap for accuracy estimation and model selection, in: Ijcai, volume vol. 14, Montreal, Canada, pp. 1137–1145.
[47] D.M. Hawkins, S.C. Basak, D. Mills, Assessing model fit by cross-validation, J. Chem. Inf. Comput. Sci. 43 (2003) 579–586.
[48] B. Yin, K. Liew, Machine learning and materials informatics approaches for evaluating the interfacial properties of fiber-reinforced composites, Compos. Struct. 273 (2021), 114328.
[49] M. Kuhn, K. Johnson, et al., Applied Predictive Modeling, vol. 26, Springer, 2013.
[50] J. Shao, Linear model selection by cross-validation, J. Am. Stat. Assoc. 88 (1993) 486–494.
[51] D.H. Wolpert, W.G. Macready, No free lunch theorems for optimization, IEEE Trans. Evol. Comput. 1 (1997) 67–82.
[52] B. Liu, N. Vu-Bac, T. Rabczuk, A stochastic multiscale method for the prediction of the thermal conductivity of polymer nanocomposites through hybrid machine learning algorithms, Compos. Struct. 273 (2021), 114269.
[53] B. Liu, N. Vu-Bac, X. Zhuang, X. Fu, T. Rabczuk, Stochastic full-range multiscale modeling of thermal conductivity of polymeric carbon nanotubes composites: a machine learning approach, Compos. Struct. (2022), 115393.
[54] J.H. Friedman, Multivariate adaptive regression splines, Ann. Stat. (1991) 1–67.
[55] L. Breiman, J. Friedman, C.J. Stone, R.A. Olshen, Classification and Regression Trees, CRC press, 1984.
[56] L. Breiman, Bagging predictors, Mach. Learn. 24 (1996) 123–140.
[57] L. Breiman, Random forests, Mach. Learn. 45 (2001) 5–32.