Transfer prior knowledge from surrogate modelling: A meta-learning approach

Minghui Cheng[a,*], Chao Dang[b], Dan M. Frangopol[a], Michael Beer[b,c,d], Xian-Xun Yuan[e]

[a] *Department of Civil and Environmental Engineering, ATLSS Engineering Research Center, Lehigh University, 117 ATLSS Dr., Bethlehem, PA 18015-4729, United States*
[b] *Institute for Risk and Reliability, Leibniz University Hannover, Callinstr. 34, Hannover 30167, Germany*
[c] *Institute for Risk and Uncertainty, University of Liverpool, Liverpool L69 7ZF, United Kingdom*
[d] *International Joint Research Center for Engineering Reliability and Stochastic Mechanics, Tongji University, Shanghai 200092, PR China*
[e] *Department of Civil Engineering, Ryerson University, Toronto, ON, M5B 2K3, Canada*

Abstract

Surrogate modelling has emerged as a useful technique to study complex physical and engineering systems in various disciplines, especially for engineering analysis. Previous studies mostly focused on developing new surrogate models and/or applying existing surrogate models to practical problems. Despite the computational efficiency, the surrogate for a new task is often built from scratch and the knowledge gained from previous surrogate modelling for similar tasks is neglected. As the need for quickly modifying simulation models to reflect design changes has significantly increased, one potential solution is to utilize prior knowledge from surrogate modelling. In this study, a novel meta-learning-based surrogate modelling framework is presented. The framework includes two phases: a meta-training and a few-shot learning phase. A meta-model that represents a family of tasks and the adaptation of this model to a new task with few data points are the results of the first and second phase, respectively. The study specifies the scope of the framework by classifying similar tasks. Applications of the framework to global sensitivity analysis, optimization, and reliability analysis are also addressed. Four numerical experiments are performed to demonstrate the feasibility and applicability of the framework.

Keywords: Meta-learning-based surrogate modelling, Model-agnostic meta-learning, Knowledge transfer, Surrogate modelling

*Corresponding author
Email address: mic316@lehigh.edu (Minghui Cheng)

1. Introduction

With the rapid development of computing power over the last few decades, computer simulations have become popular to model the physical and engineering systems. For many real-world problems, high-fidelity simulations are required to accurately predict the behavior of systems. Furthermore, in engineering analysis very often computationally intensive simulations need to be repeated many times with respect to different design configurations, so as to meet given performance requirements. To accelerate these simulation-based analyses, surrogate modelling that can mimic the behavior of the simulation model while being computationally cheaper to evaluate is gaining popularity recently in various domains, such as uncertainty quantification [1–6], engineering design optimization [7–11], and optimization algorithms [12–15]. Various types of surrogate modelling techniques, such as neural networks [16–27], polynomial chaos expansions [28–31], support vector machine [32, 33], Gaussian processes [34–38] and response surface methods [39, 40], among others, have been developed and used in recent studies. These existing studies have succeeded in reducing the number of calls to the computationally expensive simulations, which is the goal of surrogate modelling [1, 12]. However, the procedure for building a surrogate model is often carried out from scratch and the knowledge gained from previous surrogate modelling of similar tasks is ignored. The purpose of this paper is to investigate the process of transferring the prior knowledge of surrogate modelling to a new task under certain conditions.

Similar ideas for knowledge transfer can be found in machine learning, such as transfer learning [41, 42] and meta-learning [43]. Transfer learning aims at storing the knowledge gained from solving one task and applying it to another different but related task, while meta-learning seeks to learn the meta-knowledge from a distribution of source tasks that can rapidly generalize across new tasks. Their successful applications can be seen in areas such as computer vision and natural language processing [41–44]. However, very few studies have been done in this direction in engineering analysis [45–54]. Combined approximation approach was developed for efficient structural reanalysis when

changes in the design (e.g. changes in cross sections and removal and/or addition of members) is needed [45–47]. This approach reuses and modifies the stiffness matrix obtained from the existing analysis to accelerate reanalysis. Applications have been seen in topology optimization and sensitivity analysis [55–58]. Transfer learning was implemented for image-based damage detection of structural components. For example, VGGNet, AlexNet , and GoogLENet were used as a pre-trained deep convolutional neural network [48–50]. Portions of the pre-trained networks are retrained to accomplish their tasks with relatively high accuracy using only small data sets. A recent breakthrough in applications of knowledge transfer is the development of population-based structural health monitoring (PBSHM) [51–54]. The aim of these studies is to generalize the relations between features and labels from several members of a population of structures to new members of the population where only partial data points or no data points are available. The similarity among structures to identify a population is quantified and the domain adaptation technique is utilized for knowledge transfer within the population. Simulations and experimental case studies are also carried out to validate PBSHM and show an increase in the ability of damage localization and detection [53, 59]. Although these studies belong to different research areas of engineering, they show that transferring knowledge from previous engineering tasks to a new but similar task can improve the efficiency of solving this new task.

However, the above mentioned studies in engineering analysis are not suitable for surrogate modelling. The structural reanalysis approach needs the stiffness matrix of the structure, but surrogate modelling often deals with black-box problems, from which only inputs and outputs can be available. The transfer learning technique used in image-based damage detection relies on the fact that the shallow layers of the deep convolutional neural networks represent more generalized features such as texture and color [60, 61]. Hence, the parameters of the shallow layers are transferable across different domains of images. Yet, for surrogate modelling, neural networks are used as function approximators and are usually shallower, therefore do not satisfy the requirements to

implement the technique. Joint domain adaptation method used in PBSHM [53] is a classification algorithm, while fitting a surrogate model is usually a regression problem. Besides, to reduce the probability of negative transfer, given the target structure, the knowledge should be transferred from a structure with higher degrees of similarity. In PBSHM, the degree of similarity is quantified in terms of geometry, material, and topology [51, 52, 54]. This quantification is from the structural point of view, but may be limited from the black-box point of view. For example, the functions mapping the inputs to outputs can be the same for two distinct structures. In this scenario, knowledge transfer is possible for surrogate modelling, but not applicable for PBSHM. Overall, for knowledge transfer in surrogate modelling, it is necessary to define the similarity between modelling tasks and implement the appropriate algorithm to transfer prior knowledge.

This study presents a meta-learning-based surrogate modelling (MLSM) framework (the term "meta-learning" is borrowed from machine learning). To the best of the authors' knowledge, this is the first work to realize the concept of knowledge transfer for surrogate modelling in engineering analysis. The procedure of MLSM involves establishing a meta-model representing a family of modelling tasks and constructing the surrogate model for a new modelling task by adapting the meta-model. Specifically, model-agnostic meta-learning algorithm (MAML) [62] is implemented to train the meta-model. Similarity between modelling tasks is defined from the perspective of functional forms. Based on the definition, the scope of the framework is identified. Applications to global sensitivity analysis, optimization, and reliability analysis are outlined. Numerical examples are performed to illustrate the framework and show its computational efficiency.

The remainder of the paper is organized as follows. Section 2 presents an overview of the MLSM framework and discusses its mathematical formulation. Section 3 introduces the algorithm for the framework. The definition of similar tasks within the framework is provided in Section 4 and the applications are outlined in Section 5. Four numerical examples are studied in Section 6. The limitations and future work are discussed in

Section 7. Finally, the conclusions are drawn in Section 8.

2. Meta-learning-based surrogate modelling (MLSM)

2.1. Motivation

Traditionally, surrogate modelling is carried out from scratch when a new task arises. For example, the new task can be a structural analysis task, an inspection planning task, or a sensor placement task, among others. The surrogate model is fitted using the data points from the true simulation model of the task (e.g. finite element model). However, this procedure neglects the prior knowledge gained from previous tasks similar to the new one. In general, tasks are similar when they share the same set of inputs and outputs and differ in values on a limited number of parameters. For example, a new task is to find the maximum horizontal displacements of a 10-story building subjected to seismic loads. The inputs of surrogate modelling are the external loads and the outputs are the maximum horizontal displacements of all 10 stories. Any existing tasks that conducted the same analysis on a 10-story building with different configurations (e.g. different cross-sections of the columns, different material strength, and different heights) are similar to this new task from the perspective of surrogate modelling. A detailed definition of similar tasks will be given in Section 4. Chances are that utilizing the knowledge from the similar tasks is beneficial for the new task, therefore, MLSM is proposed. Figure 1 shows the comparison between conventional surrogate modelling and MLSM. For MLSM, the prior knowledge from existing similar tasks is extracted to build a meta-model. The meta-model is then combined with the data points from the true model to build the surrogate model. The effect of MLSM is the reduction of the number of data points needed to accurately build the surrogate model for the new task.

It should be noted that in the previous studies related to surrogate modelling, the terms meta-model and surrogate model are usually used interchangeably. However, in this paper, they should be distinguished. Surrogate model is the function approximation between inputs and outputs of the true model, while meta-model is considered as a

higher-level modelling of surrogate models.

Building the meta-model needs a database of existing tasks. This inevitably requires additional time to acquire data points of the tasks. However, once the meta-model is obtained, it can be stored for future use. If a large number of new tasks need to be evaluated, reusing the meta-model to build surrogate models is computationally efficient. Admittedly, without a database, traditional surrogate modelling is suggested. However, two plausible situations requiring databases are conceived for the application of MLSM. The first is regional seismic risk analysis in which a large number of buildings are assessed [63]. A database of multiple classes of buildings (e.g. different material and number of stories) can be established in advance and each class can be represented by a meta-model. When regional seismic analysis is conducted, buildings can be first grouped into classes. Knowledge can then be transferred from the meta-model to the corresponding class of buildings. The large number of buildings justifies the establishment of a database. Besides, this database can be also reused in different cities, which further rationalizes MLSM. The other is scenario-based resilience analysis. The uncertainties associated with resilience analysis can be considered via scenario analysis [64]. After numerous scenarios are analyzed by traditional surrogate modelling, a meta-model can be built based on these completed analyses. The analysis of a new scenario can be accelerated by utilizing the knowledge of the meta-model.

2.2. Mathematical formulation

The previous section introduces the general concept of MLSM. A detailed description of the MLSM framework is provided in this section. As shown in Figure 2, the framework involves (a) the meta-training phase to build the meta-model and (b) the few-shot learning phase to adapt from the meta-model for a new task. The mathematical formulation and notations of the framework are introduced as follows.

Consider a family of functions $f(\boldsymbol{X}|\boldsymbol{A})$ and $f'(\boldsymbol{X}|\boldsymbol{A}')$, where $\boldsymbol{X} = (x_1, x_2, ..., x_l)$ is the input of the function, $\boldsymbol{A}$ and $\boldsymbol{A}'$ are the coefficients of the true functions f and surrogate functions f', respectively. Suppose N tasks were already analyzed (i.e. T_1 to

T_N) within the family and N surrogate models in the database, denoted as $f'(\boldsymbol{X}|\boldsymbol{A}'_1)$ to $f'(\boldsymbol{X}|\boldsymbol{A}'_N)$, were established. For Task $T_i \in (T_1, T_2, ..., T_N)$, there exists a limited and finite number of data points, within which the jth input-output pair is denoted as $\boldsymbol{D}_{i,j} = (\boldsymbol{X}_{i,j}, \boldsymbol{Z}_{i,j})$, where $\boldsymbol{Z}_{i,j} = f(\boldsymbol{X}_{i,j}|\boldsymbol{A}_i)$ is the output of task. When the surrogate model can accurately predict outputs of the task, then $\boldsymbol{Z}_{i,j} \approx f'(\boldsymbol{X}_{i,j}|\boldsymbol{A}'_i)$

The objective of MLSM is twofold. The first is to develop a meta-model $f'(\boldsymbol{X}|\boldsymbol{A}'_{mt})$ that represents $f(\boldsymbol{X}|\boldsymbol{A}_1)$ to $f(\boldsymbol{X}|\boldsymbol{A}_N)$. The second one is to use $f'(\boldsymbol{X}|\boldsymbol{A}'_{mt})$ to assist in the development of surrogate models $f'(\boldsymbol{X}|\boldsymbol{A}'_w)$ for any new task within the family, where $T_w \notin (T_1, T_2, ..., T_N)$. With $f'(\boldsymbol{X}|\boldsymbol{A}'_{mt})$, the number of data points needed for approximating $f(\boldsymbol{X}|\boldsymbol{A}_w)$ is expected to be less than without using $f'(\boldsymbol{X}|\boldsymbol{A}'_{mt})$.

The general idea of building the meta-model is that $\boldsymbol{A}'_{mt}$ can quickly converge to $\boldsymbol{A}_w$ with very few training data points $\boldsymbol{D}_w$. This procedure is accomplished by minimization of the loss function. For surrogate modelling in this study, the mean squared error is selected as the loss function expressed as

$$\mathcal{L} = [f(\boldsymbol{X}|\boldsymbol{A}_w) - f'(\boldsymbol{X}|\boldsymbol{A}'_w)]^2 \tag{1}$$

where $f'(\boldsymbol{X}|\boldsymbol{A}'_w)$ is the surrogate model adapted from $f'(\boldsymbol{X}|\boldsymbol{A}'_{mt})$ with K data points from task T_w. The process of adapting from $f'(\boldsymbol{X}|\boldsymbol{A}'_{mt})$ to $f'(\boldsymbol{X}|\boldsymbol{A}'_w)$ with very few points can be considered as few-shot learning. Since K data points are used, it is also called K-shot learning [41, 62]. The true $\boldsymbol{A}_w$ is unknown. Therefore, to evaluate the accuracy of K-shot learning, testing data points $\boldsymbol{D}_{w.te}$ can be sampled from task T_w to evaluate the mean squared error using Eq. (1).

3. Algorithms for the framework

The MLSM framework has a meta-training phase and a few-shot learning phase, as shown in Figure 2. For the meta-training phase, MAML [62] is used to obtain the meta-model, which represents the knowledge learned from the existing tasks, while for the few-shot learning phase, surrogate models are adapting from the meta-model by training

on data points sampled from new tasks. For traditional surrogate modelling, when a new task arrives, the surrogate model is first randomly initialized and then trained to approximate the true model with the available data points. On the contrary, for MLSM, the surrogate model is first initialized as the meta-model and then trained. Hence, the process of knowledge transfer involves extracting the knowledge (i.e. building the meta-model from existing tasks using MAML) and transferring the knowledge (i.e. using it as the initialization for all the new tasks).

To be compatible with MAML, the neural network is selected as the type for both surrogate modelling and meta modelling in the framework. Besides, in engineering analysis, it has been shown that neural networks are capable of solving forward and inverse problems and tackling uncertainty quantification [16, 18, 26, 27]. The meta-model parameterized by $\boldsymbol{\theta}'_{mt}$ and the surrogate model for new task T_w parameterized by $\boldsymbol{\theta}'_{w}$ are denoted as $f'_{\boldsymbol{\theta}'_{mt}}$ and $f'_{\boldsymbol{\theta}'_{w}}$, respectively. The goals of the meta-training phase and few-shot learning phase are to find the optimal neural network parameters $\boldsymbol{\theta}'_{mt}$ and $\boldsymbol{\theta}'_{w}$, respectively.

To obtain the optimal $\boldsymbol{\theta}'_{mt}$, each epoch of MAML consists of the following four steps.

(a) Sample N_s tasks $(T_{s1}, T_{s2}, ..., T_{sn})$ from the existing tasks. For each sampled task, N_a data points are sampled from the existing data points for adapting the meta-model. This set of data points is for "inner loop update process" [65].

(b) For $T_{si} \in (T_{s1}, T_{s2}, ..., T_{sn})$, the surrogate model $f'_{\boldsymbol{\theta}'_{si}}$ is adapted from $f'_{\boldsymbol{\theta}'_{mt}}$ using the data points. Specifically, $\boldsymbol{\theta}'_{si}$ is updated using one gradient descent step

$$\boldsymbol{\theta}'_{si} = \boldsymbol{\theta}'_{mt} - \alpha_i \nabla_{\boldsymbol{\theta}'_{mt}} \mathcal{L}_{T_{si}}(f'_{\boldsymbol{\theta}'_{mt}}) \tag{2}$$

where α_i is the step size of the inner loop, and $\mathcal{L}_{T_{si}}(f'_{\boldsymbol{\theta}'_{mt}})$ is the loss function associated with T_{si} using the N_a data points. In Eq. (2), the loss function is the mean squared error between the outputs of data points and predicted values from $f'_{\boldsymbol{\theta}'_{mt}}$, as expressed in Eq. (1).

(c) For each sampled task T_{si}, N_m data points are sampled from existing data points

for optimizing the meta-model. Usually, $N_a = N_m$ are small and two sets of data points should be different. This set of N_m data points is for the purpose of "outer loop update process" [65]. The loss associated with N_m data points $\mathcal{L}_{T_{si}}(f'_{\boldsymbol{\theta}'_{si}})$ is calculated.

(d) The meta-loss is the accumulation of losses associated with all sampled tasks

$$\mathcal{L}_{mt} = \sum_{T_{si}} \mathcal{L}_{T_{si}}(f'_{\boldsymbol{\theta}'_{si}}) \tag{3}$$

After computing $\mathcal{L}_{mt}$, a gradient descent step is performed on $\boldsymbol{\theta}'_{mt}$

$$\boldsymbol{\theta}'_{mt} \leftarrow \boldsymbol{\theta}'_{mt} - \alpha_o \nabla_{\boldsymbol{\theta}'_{mt}} \mathcal{L}_{mt} \tag{4}$$

where α_o is the step size of the outer loop. A large number of epochs are run until $\mathcal{L}_{mt}$ converges. The objective of meta-traing is to find $\boldsymbol{\theta}'_{mt}$ that minimizes $\mathcal{L}_{mt}$. The procedure is shown in Algorithm 1 (adopted from [62]).

Algorithm 1: MAML for meta-training phase of the framework (adopted from [62])

```
Initialize θ'_mt;
while not done do
    Sample N_s tasks (T_s1, T_s2, ..., T_sn);
    for all T_si do
        Sample N_a data points from existing data points;
        Evaluate f'_θ'_mt;
        Compute θ'_si using Eq. (2);
        Sample N_m data points from existing data points to compute L_Tsi(f'_θ'_si);
    end
    Compute meta-loss L_mt using Eq. (3);
    Perform an update on θ'_mt using Eq. (4);
end
```

The ultimate goal of surrogate modelling is to reduce the number of data points needed to build a relatively accurate surrogate model. MAML achieves this goal by explicitly optimizing the meta-model to be capable of adapting to new tasks quickly with small data sets. Specifically, this ability is quantified by minimizing $\mathcal{L}_{mt}$. In fact,

the meta-model obtained using MAML algorithm is a good initialization for new tasks. Therefore, a small number of data points is possible to adapt from the meta-model to the model representing the new task. There exists numerous meta-learning algorithms. Some are variants of MAML, such as Reptile [66], MAML++ [65], and iMAML [67], while other aspects to accelerate meta-learning are focused, such as distribution calibration [68] and learning the loss functions [69]. Although the novelty of this paper is to propose the MLSM framework, future work can be conducted to compare and benchmark meta-learning algorithms for surrogate modelling.

The procedure for few-shot learning phases includes (a) sampling data points from the new task and (b) optimizing the surrogate model using the data points. When optimizing $\boldsymbol{\theta}'_w$, the first update is an update from $\boldsymbol{\theta}'_{mt}$, which represents the adaptation from the meta-model

$$\boldsymbol{\theta}'_w = \boldsymbol{\theta}'_{mt} - \alpha_i \nabla_{\boldsymbol{\theta}'_{mt}} \mathcal{L}_{T_w}(f'_{\boldsymbol{\theta}'_{mt}}) \tag{5}$$

while later updates using the same data points are

$$\boldsymbol{\theta}'_w \leftarrow \boldsymbol{\theta}'_w - \alpha_i \nabla_{\boldsymbol{\theta}'_w} \mathcal{L}_{T_w}(f'_{\boldsymbol{\theta}'_w}) \tag{6}$$

It should be noted that any gradient descent methods can be used for all the updates in the meta-training and few-shot learning phases. In this paper, stochastic gradient descent is used for Eq. (2), while the Adam optimizer [70] is used for other updates.

4. Scope of the framework

To implement the framework for engineering problems, it is essential to know what can be categorized as similar tasks. In this study, it is proposed that a task T_i is similar to another task T_j if the following two requirements are met: (a) $\boldsymbol{X}_i$ and $\boldsymbol{X}_j$ are the same; and (b) $\boldsymbol{A}_i$ and $\boldsymbol{A}_j$ can be substituted by each other.

Several examples are given to explain the above two requirements by considering the

following equations

$$f(x_1, x_2, x_3|a_1, a_2, a_3) = \sin(x_1) + \frac{a_2 x_2}{a_1 x_1 + x_3} - a_3 x_3 \tag{7}$$

$$f(x_1, x_2, x_3|a_1, a_2, a_4, a_5) = \sin(x_1) + \frac{a_2 x_2}{a_1 x_1 + x_3} - \sqrt{a_4^2 + a_5^2} x_3 \tag{8}$$

$$f(x_1, x_2, x_3, x_4|a_1, a_2, a_3) = \sin(x_1) + \frac{a_2 x_2}{a_1 x_1 + x_3} - a_3 x_3 - x_4 \tag{9}$$

$$f(x_1, x_2, x_3|a_1, a_2, a_6) = \sin(x_1) + \frac{a_2 x_2}{a_1 x_1 + x_3} - a_6 x_3^2 \tag{10}$$

$$f(x_1, x_2, x_3|a_1, a_2, a_3, a_7) = \sin(x_1 + a_7) + \frac{a_2 x_2}{a_1 x_1 + x_3} - a_3 x_3 \tag{11}$$

reflecting the true models of the tasks. Consider the task described by Eq. (7) as the baseline task. Overall, the task described by Eq. (8) is similar to the baseline task, while tasks described by Eqs. (9) to (11) are different. Specifically, the task described by Eq. (8) is similar to the baseline task because a_3 in Eq. (7) and $\sqrt{a_4^2 + a_5^2}$ in Eq. (8) can be interchanged by each other. The task described by Eq. (9) is different because it has an additional input x_4. Although the task described by Eq. (10) has the same inputs as the tasks described by Eqs. (7) to (8), it has the term x_3^2 with the coefficient a_6 that cannot be substituted, thus violating the second requirement. Special attention should be paid to Eqs. (11) and (7). Though Eq. (7) can be derived from Eq. (11) by assigning $a_7 = 0$, a_7 cannot be substituted by a_1 to a_3. Hence, by definition, these two tasks are not similar.

An engineering problem is used to illustrate the practical meaning of the above two requirements. Consider risk-based inspection planning of a fatigue detail as an example. The objective is to find the management plan that minimizes the expected life-cycle cost. For this example, the set of decision variables $\boldsymbol{X}$ includes the timings of two inspections and the maintenance threshold of the crack size. The set of coefficients $\boldsymbol{A}$ can include any parameters and metrics related to the inspection planning. For a baseline task, $\boldsymbol{A}$ includes the cost metrics (i.e. costs of maintenance and failure) and the failure criterion

(i.e. the failure threshold of the crack size). The cost metrics impact the expected life-cycle cost explicitly, while the failure threshold of the crack size implicitly changes that by modifying the failure probability profile. Any tasks that modify the values of these parameters in $\boldsymbol{A}$ are similar to the baseline task. It is more complicated if parameters associated with crack growth, such as stress ranges and the annual number of cycles, are changed. Similar to the failure threshold, these parameters implicitly affect the expected life-cycle cost by modifying the structural performance profile. On the one hand, they cannot directly substitute the failure threshold. Therefore, based on the definition of similar tasks, tasks that changes the parameters associated with crack growth are not similar to the baseline task. On the other hand, slightly altering the values of these parameters may have comparable effects on expected life-cycle cost as that of the failure threshold. Hence, chances are that the knowledge learned from tasks similar to the baseline task can be transferred to tasks that slightly altering the structural performance and failure probability profile. The above discussion will be demonstrated in detail in section 6.4. However, tasks that modify $\boldsymbol{X}$, such as optimizing the timings of three instead of two inspections, are different tasks.

Although the concept of similar tasks is defined from the perspective of functional forms, it is not necessary for the family of tasks to have explicit mathematical expressions. Consider the example of the 10-story building discussed in section 2.1. The computation of maximum horizontal displacements of 10 stories is usually so complicated that it is hard to derive an explicit mathematical expression. For risk-based inspection planning, which is mentioned previously, life-cycle cost cannot be explicitly expressed as a function of inspection timings and the maintenance threshold.

With the definition of similar tasks, the scope of the framework can be defined accordingly. A family of tasks is a set of tasks that are similar. The existing tasks for meta-training must belong to one family. The obtained meta-model can be used to learn any new tasks within the family. However, it is not necessary for the new task to be in the family so that the meta-model can be used. If a new task is derived from a task of

the family by assigning specific values to some of its coefficients, then it can be learned by the meta-model. For example, the meta-model trained from the family that includes tasks described by Eq. (11) can be used to learn the task described by Eq. (7). Yet, the reverse doesn't apply.

5. Applications of the framework

The direct application of the framework is to make prediction. Given a new task T_w with the true function $f(\boldsymbol{X}|\boldsymbol{A}_w)$, a surrogate model $f'(\boldsymbol{X}|\boldsymbol{A}'_w)$ is obtained and used for prediction. With the ability of global approximation, the framework can be applied to analyses related to uncertainty quantification such as probability-boxes propagation [71] and stochastic model updating [72]. In this paper, the focus is on global sensitivity analysis, optimization, and reliability analysis. For the first two, the procedures are the same as prediction until obtaining the surrogate models. Since the computation time of a surrogate model is minimal, global sensitivity analysis can be carried out in three steps: (a) sampling inputs $\boldsymbol{X}_g$, (b) evaluating $f'(\boldsymbol{X}_g|\boldsymbol{A}'_w)$, and (c) compute Sobol's indices. Sobol's indices are indicators for variance-based global sensitivity analysis. They can reflect the contributions to the uncertainty of the output from individual input parameters and their combinations [28]. Herein, the python package SALib [73] is used to sample $\boldsymbol{X}_g$ and calculate the Sobol's indices. Solving optimization includes two steps: (a) finding the inputs $\boldsymbol{X}'_o$ that optimizes $f'(\boldsymbol{X}|\boldsymbol{A}'_w)$, and (b) evaluating $f(\boldsymbol{X}'_o|\boldsymbol{A}_w)$ as the optimal value of the true function.

Reliability analysis is formulated differently within the framework. In general, random variables, which are the inputs of the limit state functions, follow certain kinds of distributions. Based on the distribution, each random variable has its descriptors, such as mean μ and coefficient of variation (COV) δ. With the same mathematical expressions, a task is similar to other tasks if only the descriptors of the random variables change. This is a realistic situation in practice. For example, if the reliability of using C30 concrete is not satisfactory, then the reliability of using C40 concrete may need to

be evaluated. However, $f(\boldsymbol{X}|\boldsymbol{A})$ cannot reflect the changes in the descriptors. Besides, during the meta-training phase, $\boldsymbol{X}$ is sampled over a set of prescribed ranges. This is not feasible with $\boldsymbol{X}$ having different sets of mean values across different tasks. For the above two reasons, normalization is needed. Suppose $\boldsymbol{X}$ has sets of descriptors $\boldsymbol{M}$ and $\boldsymbol{\Delta}$, where $\boldsymbol{M}$ and $\boldsymbol{\Delta}$ stores the mean values and COV of all the elements in $\boldsymbol{X}$, respectively. $f(\boldsymbol{X}|\boldsymbol{A})$ is rewritten as $f(\boldsymbol{Y}|\boldsymbol{A},\boldsymbol{M})$, where each element of $\boldsymbol{Y}$ is obtained by normalizing the corresponding element in $\boldsymbol{X}$ by its mean. As a result, $\boldsymbol{Y}$ has a unit mean and COV of $\boldsymbol{\Delta}$.

Use Eq. (7) as an example. x_1, x_2, and x_3 of an existing task follow normal distributions with means and COVs as (μ_1, δ_1), (μ_2, δ_2), and (μ_3, δ_3), respectively. x_{w_1}, x_{w_2}, and x_{w_3} of the new task also follow normal distributions, but with different means and same COV as (μ_{w_1}, δ_1), (μ_{w_2}, δ_2), and (μ_{w_3}, δ_3), respectively. To consider this situation, Eq. (7) is rewritten to comply to the standard format of the framework. For the existing task, it is normalized as

$$f(y_1, y_2, y_3|a_1, a_2, a_3, \mu_1, \mu_2, \mu_3) = \sin(\mu_1 y_1) + \frac{a_2\mu_2 y_2}{a_1\mu_1 y_1 + \mu_3 y_3} - a_3\mu_3 y_3 \tag{12}$$

where y_1, y_2, and y_3 of the existing task follow normal distributions with means and COVs as $(1, \delta_1)$, $(1, \delta_2)$, and $(1, \delta_3)$, respectively. For the new task, it is normalized as

$$f(y_1, y_2, y_3|a_1, a_2, a_3, \mu_{w_1}, \mu_{w_2}, \mu_{w_3}) = \sin(\mu_{w_1} y_1) + \frac{a_2\mu_{w_2} y_2}{a_1\mu_{w_1} y_1 + \mu_{w_3} y_3} - a_3\mu_{w_3} y_3 \tag{13}$$

It is observed that $y_1 = x_1/\mu_1 = x_{w_1}/\mu_{w_1}$, $y_2 = x_2/\mu_2 = x_{w_2}/\mu_{w_2}$, and $y_3 = x_3/\mu_3 = x_{w_3}/\mu_{w_3}$. Based on the definition of similar tasks, the mathematical expressions of the new task and existing task are the same. Therefore, the new task can be approximated through few-shot learning. The obtained surrogate model can then be used to calculate the failure probability by Monte Carlo simulation. It should be noted that when calculating the failure probability, sampling of $\boldsymbol{Y}$ follows its distribution, while during the meta-training phase, $\boldsymbol{Y}$ is sampled uniformly.

6. Numerical examples

Four numerical examples are carried out to demonstrate the computational efficiency of the framework. Comparisons are made between two scenarios:

- Traditional surrogate modelling (TSM): the surrogate model is constructed from scratch without the assistance of the meta-model.
- MLSM: the surrogate model is constructed with the assistance of the meta-model.

The focus of comparisons is on the number of data points required to achieve a target approximating accuracy for a new task. Specifically, there exists two approaches for comparisons: (a) having the same number of data points, if the accuracy is better, the technique is better; (b) to achieve the same accuracy, if the required number of data points is smaller, the technique is better. Comparisons in examples 1 and 2 are made using the first approach, while those in examples 3 and 4 are made using the second approach. The sampling method for both scenarios is simple random sampling.

For MLSM, it is assumed that the size of the database of existing tasks is adequate to obtain the meta-model. Particularly, the database has 200 meta-training tasks and for each task, 500 data points (i.e. input-output pairs) are available for examples 1 to 3. Creating a database of tasks before the meta-training phase is done by first sampling 200 sets of coefficients of the true function. Then for each task within the database, 500 inputs are sampled and the corresponding outputs are evaluated. For example 4, the database has 100 tasks with 200 data points for each task. The procedure to build the database is the same as that used in examples 1 to 3. The ability of knowledge transfer of the meta-model is evaluated on at least four testing tasks for each numerical example. It should be noted that the meta-training and few-shot learning tasks are based on existing numerical examples. To generate the large number of tasks in this study, the procedure involves (a) selecting coefficients or parameters from those of the original numerical examples and (b) altering their values around the original values (e.g.

multiply or divide the original value of a coefficient by 2). When generating the tasks, the testing tasks are excluded from the meta-training tasks in the database.

During the meta-training phase, the hyperparameters are $N_s = 25$, $N_a = N_m = 10$, $\alpha_i = 0.01$, and $\alpha_o = 0.001$. The meta-training process has 30,000 epochs. The neural network architectures for both TSM and MLSM are the same. More details about the architectures can be found in Appendix.

6.1. Example 1: Illustration of meta-learning

Example 1 is adapted from McCormick function [74]. The function is expressed as

$$f(x_1, x_2) = a_1 \sin(x_1 + x_2) + (x_1 - x_2)^2 + a_2 x_1 + a_3 x_2 \tag{14}$$

where x_1 and x_2 are between -3 and 3. For meta-training, a_1, a_2, and a_3 varies from 0.5 to 3.5, from -1.8 to -1.2, and from 2.2 to 2.8, respectively. Four testing tasks are considered. Task I is the original McCormick function where $a_1 = 1$, $a_2 = -1.5$, and $a_3 = 2.5$. Task II is when $a_1 = 4$, $a_2 = -1.5$, and $a_3 = 2.5$. Task III is when $a_1 = 4$, $a_2 = -2$, and $a_3 = 2$. Task IV is described by the following equation

$$f(x_1, x_2) = a_1 \sin(x_1 + x_2 + a_4) + (x_1 - x_2)^2 + a_2 x_1 + a_3 x_2 \tag{15}$$

where $a_1 = 1$, $a_2 = -1.5$, $a_3 = 2.5$, and $a_4 = \pi/2$. Specifically, Tasks II and III are intentionally selected because their coefficients are not among the ranges of existing tasks. Tasks I to III belong to one family, while Task IV doesn't because this task has an additional coefficient a_4. For each task, 5 runs are performed to construct the surrogate model and the losses are computed using Eq. (1). Then the run with the median value among 5 losses is selected.

Figure 3 shows the contour plots of the meta-model. Figures 4, 5, and 6 shows the comparisons of contour plots of the surrogate models between TSM and MLSM for Tasks I to III, respectively. Generally, the shape of McCormick function has two characteristics. First, the term $(x_1 - x_2)^2$ results in a U shape, which can be seen from

the top left to the bottom right of the contour plots of the true function. Second, the valley of the function surface can be seen from the top right to the bottom left. These two parts dictate the approximation accuracy. For TSM, great accuracy cannot be achieved for both parts simultaneously due to the small number of data points. On the contrary, the meta-model has already learned the overall characteristics of McCormick function, as shown in Figure 3. With very few points, the meta-model can quickly adapt to the new tasks. Figures 5 and 6 show that even if the new task is not within existing tasks, the meta-model can still quickly adapt to approximate the new task. a_1 is larger for Tasks II and III. As a result, more fluctuations are included, as shown by the curvy contours in Figures 5a and 6a. For TSM, it is difficult to approximate the fluctuations with very few data points, while the approximation results are better with the help of the meta-model.

Table 1 shows the approximation accuracy quantitatively. For illustrative purposes, a large number of data points are sampled to calculate the mean squared error. For TSM, when the available data points of the new task (i.e. $K = 10$ and $K = 20$) are scarce, the approximation precision is poor, while, for MLSM, the precision is better. Particularly, for Tasks I to III, the accuracy of MLSM with $K = 10$ is much better than that of TSM with $K = 20$, while the accuracy of MLSM with $K = 20$ is slightly better than that of TSM with $K = 40$. From Tasks I to III, more fluctuations occur. This leads to decrease in the approximation accuracy for both TSM and MLSM. Nonetheless, the approximation accuracy of MLSM is better. The difference between Tasks I and IV is the first term. This will only influence the fluctuations, therefore, approximation precisions of Tasks I and IV are comparable. Comparison shows that the approximation accuracy are relatively the same for TSM, while for MLSM, the accuracy of Task IV is worse than that of Task I. This implies that precision drops if the new task is not within the family of existing tasks. Hence, it is essential to classify tasks.

6.2. *Example 2: Application to global sensitivity analysis*

Example 2 is adapted from Ishigami function [28]. The function is expressed as

$$f(x_1, x_2, x_3) = \sin(x_1) + a_1 \sin^2(x_2) + a_2 x_3^4 \sin(x_1) \tag{16}$$

where x_1 to x_3 range between $-\pi$ to π. For meta-training, a_1 and a_2 varies from 3.5 to 35 and from 0.05 to 0.5, respectively. Five testing tasks are considered. The coefficients of the tasks are shown in Table 2. Coefficients of Tasks I (i.e. the original Ishigami function in [28]) and II are within the range of those of existing tasks, while those of Tasks III to V are outside the range. It should be noted that as a_1 and a_2 increase, more fluctuations occur, which makes it harder to achieve good approximation precision.

Table 2 also shows the approximation accuracy of the five tasks. The mean squared errors are calculated as in example 1. For Task I, the mean squared error of MLSM with $K = 60$ is smaller than that of TSM with $K = 120$. For Tasks II to IV, mean squared errors of MLSM with $K = 30$ are smaller than those of TSM with $K = 120$. This shows that MLSM can achieve better accuracy with 50% of the data points used by TSM. However, for Task V, there is no improvement of approximation accuracy with the meta-model. Please note that mean squared errors cannot be compared across different tasks because the amplitudes of the function values are different.

Global sensitivity analysis is then carried out for five tasks using the surrogate models adapted from the meta-model. Tables 3 to 7 show the Sobol's indices for Tasks I to V, respectively. In each table, Sobol's indices of the meta-model, true model, and two cases (i.e. $K = 30$ and $K = 60$) of the corresponding task are compared. Since Eq. (16) has only three terms, among all the first order indices, only S_1, S_2, and S_{13} are shown. It should be noted that for all tasks, the same meta-model is used as the initialization. Therefore, the Sobol's indices for the meta-model in Tables 3 to 7 are the same. Comparing the indices between the meta-model and the true model can reveal their similarities. For Tasks I and II, 30 data points are enough to obtain indices that are closed to those of the true models with the maximum absolute error 0.05. For Tasks

III and IV, 60 data points can obtain indices relatively closed to true values with the maximum absolute error 0.11. However, for Task V, the indices drastically deviate from the true indices. This implies that the surrogate model cannot estimate the function globally. The accuracy of calculating indices may be attributed to similarity between the meta-model and the true models of new tasks. Tables 3 and 4 show that the true indices of Tasks I and II are closed to those of the meta-model, therefore a small amount of data points is adequate. For Tasks III and IV, the true indices change significantly from those of the meta-model, hence more data points are needed.

It can be seen from Tables 3 to 7 that changes of a_1 and a_2 can notably change Sobol's indices. This indicates that the shapes of Ishigami functions vary significantly. Plus, the sinusoidal terms of Ishigami function complicates the development of surrogate models. Therefore, the adaptation from the meta-model may not achieve good precision for certain tasks drastically deviating from the meta-model (e.g. Task V). However, as shown by Tasks I to IV, the meta-model is still a good initialization for establishing the surrogate models. Therefore, MLSM shows great potential to transfer prior knowledge to new tasks.

6.3. Example 3: Application to reliability analysis

Example 3 involves a nonlinear oscillator, as shown in Figure 7. The performance function, which is adapted from the example studied in [18, 75], is

$$g(x_{k_1}, x_{k_2}, x_{f_1}, x_{t_1}, x_m, x_r) = 3x_r - \left| \frac{2x_{f_1}}{x_m(x_{k_1} + x_{k_2})} \sin\left(\frac{x_{t_1}}{2} \sqrt{\frac{x_{k_1} + x_{k_2}}{x_m}} \right) \right| \tag{17}$$

where x_{k_1}, x_{k_2}, x_{f_1}, x_{t_1}, x_m, and x_r are normally distributed random variables with means and COVs of $(\mu_{k_1}, \delta_{k_1})$, $(\mu_{k_2}, \delta_{k_2})$, $(\mu_{f_1}, \delta_{f_1})$, $(\mu_{t_1}, \delta_{t_1})$, (μ_m, δ_m), and (μ_r, δ_r), respectively. The first term in Eq. 17 represents the allowable displacement of the spring, while the second term is the maximum displacement under the pulse excitation [75]. When g is smaller than zero, the system fails. Specifically, δ_{k_1}, δ_{k_2}, δ_{f_1}, δ_{t_1}, δ_m, δ_r are 0.1, 0.1, 0.2,

0.2, 0.05, and 0.1, respectively. The performance function is rewritten as

$$f(y_{k_1}, y_{k_2}, y_{f_1}, y_{t_1}, y_m, y_r | \mu_{k_1}, \mu_{k_2}, \mu_{f_1}, \mu_{t_1}, \mu_m, \mu_r) = 3y_r\mu_r - \left| \frac{2y_{f_1}\mu_{f_1}}{y_m\mu_m(y_{k_1}\mu_{k_1} + y_{k_2}\mu_{k_2})} \sin\left(\frac{y_{t_1}\mu_{t_1}}{2} \sqrt{\frac{y_{k_1}\mu_{k_1} + y_{k_2}\mu_{k_2}}{y_m\mu_m}} \right) \right| \quad (18)$$

where y_{k_1}, y_{k_2}, y_{f_1}, y_{t_1}, y_m, and y_r are normally distributed random variables with means and COVs of $(1, \delta_{k_1})$, $(1, \delta_{k_2})$, $(1, \delta_{f_1})$, $(1, \delta_{t_1})$, $(1, \delta_m)$, and $(1, \delta_r)$, respectively.

For meta-training, μ_{k_1}, μ_{f_1}, μ_{t_1}, and μ_m varies from 0.8 to 1.2. μ_{k_2} varies from 0.08 to 0.12, while μ_r varies from 0.4 to 0.6. During meta-training, y_{k_1}, y_{k_2}, y_{f_1}, y_{t_1}, y_m, and y_r are sampled uniformly ranging from one minus five standard deviations to one plus five standard deviations. For the original task in [18], μ_{k_1}, μ_{k_2}, μ_{f_1}, μ_{t_1}, μ_m, and μ_r are 1.0, 0.1, 1.0, 1.0, 1.0, and 0.5, respectively. The corresponding failure probability P_f is 0.0283. Six testing tasks are selected with P_f ranging from 0.002 to 0.046. The information of the tasks are shown in Table 8. All tasks are not within the existing tasks. For these tasks, one to three coefficients are outside the ranges of the existing tasks.

For each task, the surrogate function $f'(y_{k_1}, y_{k_2}, y_{f_1}, y_{t_1}, y_m, y_r)$ is adapted from the meta-model with several data points. It is then used for reliability analysis. It should be noted that when conducting reliability analysis, y_{k_1}, y_{k_2}, y_{f_1}, y_{t_1}, y_m, and y_r are sampled from normal distributions. Monte Carlo simulation with 10^6 samples are used. Denote P'_f as the failure probability calculated using $f'(y_{k_1}, y_{k_2}, y_{f_1}, y_{t_1}, y_m, y_r)$. Different data points lead to different $f'(y_{k_1}, y_{k_2}, y_{f_1}, y_{t_1}, y_m, y_r)$ and eventually different P'_f. When the number of data points is small, the calculated P'_f is unstable. To evaluate the stability of surrogate modelliing, 10 runs are conducted for each task to compute the average and COV of P'_f. Starting from 10 data points, an incremental of 10 is taken to build the surrogate functions until (1) the relative error between the average P'_f and P_f is smaller than 20% and (2) COV of ten calculated P'_f are smaller than 30%. Table 8 also compares the required number of data points to achieve the targeted accuracy. With the

help of the meta-model, they are reduced by 80%, 37%, 64%, 50%, 71%, and 50%, for Tasks I to VI, respectively. The failure probability corresponding to the meta-model is 0. Although the meta-model is different from the true models of the new tasks, MLSM still outperforms TSM. This shows that the meta-model is a better initialization than the random initialization.

6.4. Example 4: Application to optimization

Example 4 involves risk-based inspection planning of fatigue details. As a ship operates through its service life, the fatigue crack can grow and the failure risk increases over time. Inspection and repair actions can be conducted to reduce the risk, but requires additional investment. Risk-based inspection planning is a systematic approach to perform optimization that minimizes expected life-cycle cost. Paris' law is used to model the crack growth [76]

$$\frac{da}{dN} = C \cdot (\sigma_r G \sqrt{\pi a})^m \tag{19}$$

where a is the crack size, N is the number of cycles, C and m are material parameters, σ_r is the stress range, and G is the geometry parameter. Solving Eq. (19) given the initial crack size a_0, the time-dependent crack size is [77]

$$a(t) = \begin{cases} a_0 \exp(CG^2\sigma_r^2 N_{avg} t), & m = 2 \\ [(1-\frac{m}{2})CG^m\sigma_r^m N_{avg} t \pi^{\frac{m}{2}} + a_0^{(1-\frac{m}{2})}]^{\frac{1}{1-m/2}}, & m \neq 2 \end{cases} \tag{20}$$

where N_{avg} is the average annual number of cycles. When the crack size $a(t)$ reaches the critical crack size for failure a_f, the fatigue detail fails.

In general, the management plan includes inspections and the possible ensuing repairs. At each inspection, the fatigue crack can be detected or not. The probability of detection is [78]

$$PoD(a) = 1 - \exp(-\frac{a}{a_\mu}) \tag{21}$$

where a_μ is the mean detectable crack size. When the crack is detected, if the crack size is larger than the critical crack size for repair a_r, then the fatigue detail is repaired. It is assumed that after repair, the detail returns to its initial condition. Given a management plan, the life-cycle cost is

$$C_{LC} = N_{ins}C_{ins} + N_{rep}C_{rep} + N_f C_f \tag{22}$$

where N_{ins}, N_{rep}, and N_f are the number of inspections, repairs, and failure, respectively, throughout the service life, and C_{ins}, C_{rep}, and C_f are the costs of an inspection, repair, and failure, respectively. For a given plan, Monte Carlo simulation is conducted to compute the distribution of life-cycle cost. Then the expected life-cycle cost $\mathbb{E}(C_{LC})$ is obtained. For this numerical example, $N_{ins} = 2$. Optimal inspection planning is formulated as

$$\begin{aligned} \text{find} \quad & t_1,\ t_2,\ a_r \\ \min \quad & \mathbb{E}(C_{LC}) \\ \text{s.t.} \quad & t_2 - t_1 \geq 3 \\ & t_1 \geq 3, \quad t_2 \leq t_{end} - 2 \\ & 1.0 \leq a_r \leq 10.0 \end{aligned} \tag{23}$$

where t_1 and t_2 are the timings of the first and second inspection, respectively, and t_{end} is the service life. Herein, t_1, t_2, and a_r are treated as discrete variables. Specifically, t_1 and t_2 are integers and a_r increments from 1.0 to 10.0 with an interval of 0.1.

A baseline scenario is first provided. The meta-training tasks and testing tasks are generated by altering the values of the parameters of the baseline scenario. The parameters of the baseline scenario is shown in Table 9. C_{rep}, C_f, and a_f are selected as the parameters to be varied to generate meta-training tasks. Particularly, C_{rep} and C_f vary from 25 to 100 and from 101 to 400, respectively, while a_f varies from 30 to 70. Changes to C_{rep} and C_f represent the explicit modification to C_{LC}, while the change to a_f implicitly alter C_{LC}. Eight testing tasks are considered. Task I is the baseline

scenario. For the other seven testing tasks, the values of one or two parameters are changed. Task II is when $C_f = 300$ and $a_f = 80$, while Task III is when $C_{rep} = 20$ and $C_f = 100$. Task IV only changes a_μ from 1.8 to 1.4, which means the probability of detection increases. Task V increases the mean value of N_{avg} from 5×10^5 to 1×10^6, while Task VI decreases G from 1.12 to 1.05. Task VII shortens t_{end} from 25 years to 23 years, while Task VIII extends it from 25 years to 27 years. Based on the definition of similar tasks, only Tasks I to III are within the family of meta-training tasks. It should be noted that the decision variable space is changed for Tasks VII and VIII. Specifically, ranges of t_1 and t_2 reduce and increase for Tasks VII and VIII, respectively.

Optimal solutions for the testing tasks, as shown in Table 10, are obtained using exhaustive search to provide benchmarks for comparisons. Following the procedure described in section 5, the surrogate model is built for each task by sampling data points and the optimal $\mathbb{E}(C_{LC})$ is acquired. As the number of data points increase, the optimal $\mathbb{E}(C_{LC})$ obtained by MLSM and TSM will both approach the true optimum. Starting from 20 data points, an incremental of 20 is taken until the relative error between the true optimum and the optimum obtained by surrogate modelling is smaller than 15%. Table 11 compares the required number of data points between TSM and MLSM. With the help of the meta-model, the required numbers of data points are reduced by 64%, 44%, 60%, 50%, 73%, 63%, and 50% for Tasks I to VII, respectively. However, the effect is minimal for Task VIII. The improvements in Tasks I to III show that MLSM can work well when the new tasks are similar to the meta-training tasks. Unlike examples 1 to 3 that have explicit expressions, $\mathbb{E}(C_{LC})$ cannot be explicitly written as a function of t_1, t_2, and a_r. This improvement also shows that MLSM is suitable even when the similar tasks are not classified from the perspective of functional forms. Since surrogate modelling generally involves black-box computations, this characteristic implies that MLSM can be applicable to various types of engineering analysis. Similar to example 3, deviation of the meta-model from the true models is observed. Specifically, the minimum $\mathbb{E}(C_{LC})$ is 37.06 and $t_1 = 3$, $t_2 = 6$, and $a_r = 10.0$. Future work is needed to explain why the

meta-model is a better initialization.

Besides, as shown by Tasks IV to VI, it is possible that MLSM can still improve the computational efficiency when the new task is not similar to the meta-training tasks. Specifically, Task IV modifies the inspection quality, while Tasks V and VI modifies the structural performance profile. This implies that the prior knowledge learned from managing one fatigue detail under different circumstances may be transferred to other fatigue details and inspection techniques. Although Tasks VII and VIII both changes the decision variable space, better performance is achieved for Task VII. This may be due to the fact that the input space of Task VII is a subset of that of meta-training tasks, but that of Task VIII is not. When generalizing to inputs that have not been seen in meta-training tasks, large prediction errors may occur due to negative transfer. This demonstrates the importance of keeping a constant decision variable space across all the tasks for MLSM.

7. Limitations

The framework in the present form has the following limitations. (a) When the function does not have much fluctuations, the effects of MLSM on reducing the required number of data points are minimal. TSM is adequate to fit the function with relatively small number of data points. (b) Similar to PBSHM [53], negative transfer is observed (i.e. Task V of example 2 and Task VIII of example 4). Empirical results in the field of computer science suggest that in general, negative transfer is related to task similarity [79–81]. However, there is very limited research on the theoretical foundation of negative transfer [82, 83]. Questions in this line of research remain open and are out of the scope of the current paper. More analytical and empirical research studies are needed to explore under what circumstances knowledge transfer has minimal or negative effects such as proposing indices to quantify task similarity. (c) The size of the database is large and fixed in this study. Future work is needed to investigate how to efficiently build a database adequate to obtain a meta-model with fast adaptability. (d) The

sampling method used in this study is simple random sampling. When the data points are scarce, the obtained surrogate model varies depending on the batch of sampling data points. This leads to instability in the results (e.g. failure probability). Future research incorporating efficient sampling methods can be conducted to accelerate and stabilize the process of MLSM. (e) MLSM can achieve a relatively high approximation accuracy with a small amount of data points. However, to further increase the precision needs a large incremental of data points, which compromises the advantages of MLSM. It is worthwhile to investigate how to utilize the ability of global approximation with few data points for further refining different types of analysis. One potential solution is to implement adaptive sampling strategy.

8. Conclusions

This paper presents a meta-learning-based surrogate modelling (MLSM) framework. The framework is the first to realize the idea of knowledge transfer for surrogate modelling. Compared to traditional surrogate modelling (TSM) method, MLSM utilizes knowledge gained from previous similar tasks to accelerate the new modelling tasks. It consists of a meta-training phase to build a meta-model that represents a family of existing tasks and a few-shot learning phase to build the surrogate model for a new task by adapting from the meta-model with few data points. MAML algorithm is adopted in MLSM to obtain the meta-model. The meta-model is optimized so that it can adapt to new tasks quickly with small data sets. The definition of similar tasks from the perspective of mathematical expressions is given to specify the scope of the framework. Specifically, two similar tasks should have same inputs and their coefficients of the functions should be substituted by each other. Based on this definition, MLSM in this study stipulates the situation when knowledge transfer can occur.

Four numerical examples are provided to demonstrate the advantages of knowledge transfer. For examples 1 and 2, smaller mean squared errors are achieved by MLSM with 50% of the data points used by TSM. For examples 3 and 4, compared to TSM, the

required numbers of data points for MLSM to achieve the target accuracy are reduced by 37% to 80%. These results show that MLSM can improve the computational efficiency by knowledge transfer.

The four examples are regression, global sensitivity analysis, reliability analysis, and optimization. This demonstrates that MLSM can have various applications. When implementing MLSM, it is essential to make sure that the existing and new tasks are in the same family, as revealed by the decrease in performance when violating the requirement (i.e. Task IV of example 1 and Task VIII of example 4). Sometimes, transformation (e.g. normalization in example 3) should be made to comply with the definition of similar tasks. Although the coefficients of new tasks are outside the ranges of those of existing tasks, computational efficiency can still be improved. The practical implication, as illustrated by example 4, is that prior knowledge can be transferred to new cost metrics, new fatigue details, and new inspection techniques. This shows the flexibility of MLSM.

Nevertheless, the proposed framework suffers from limitations, such as the occurrence of negative transfer, the need of a database, and the instability due to the lack of efficient sampling methods. Future work in this regard should be conducted to improve the current framework by reducing its limitations.

Declaration of Competing Interest

The authors declare that there is no conflict of interest.

Acknowledgements

The second author is grateful for the financial support received from China Scholarship Council (CSC).

Appendix A. Neural network architectures for numerical examples

The feedforward artificial neural networks (ANN) are used in this paper for the purpose of regression. An ANN consists of an input layer, an output layer, and hidden

layers [84]. Each node of the input layer corresponds to one input variable of the function, while the output layer has one node because of only one output value for the functions. The input and output layers are fully connected by an arbitrary number of hidden layers with arbitrary numbers of nodes. Each node receives the values from all the nodes in the previous layer and convert them to a single value through weights, bias values and the nonlinear activation function. In this paper, the sizes of input layers are 2, 3, 6, and 3 for examples 1, 2, 3, and 4, respectively. The number and the size of hidden layers are 2 and 64, respectively, which are the same for all examples. The nonlinear activation function is ReLU [85].

References

[1] B. Sudret, Meta-models for structural reliability and uncertainty quantification, arXiv preprint arXiv:1203.2062 (2012).

[2] N. E. Owen, P. Challenor, P. P. Menon, S. Bennani, Comparison of surrogate-based uncertainty quantification methods for computationally expensive simulators, SIAM/ASA Journal on Uncertainty Quantification 5 (1) (2017) 403–435.

[3] P. Wei, X. Zhang, M. Beer, Adaptive experiment design for probabilistic integration, Computer Methods in Applied Mechanics and Engineering 365 (2020) 113035.

[4] K. Cheng, Z. Lu, C. Ling, S. Zhou, Surrogate-assisted global sensitivity analysis: an overview, Structural and Multidisciplinary Optimization 61 (3) (2020) 1187–1213.

[5] R. Teixeira, M. Nogal, A. O'Connor, Adaptive approaches in metamodel-based reliability analysis: A review, Structural Safety 89 (2021) 102019.

[6] Y. Peng, T. Zhou, J. Li, Surrogate modeling immersed probability density evolution method for structural reliability analysis in high dimensions, Mechanical Systems and Signal Processing 152 (2021) 107366.

[7] G. G. Wang, S. Shan, Review of Metamodeling Techniques in Support of Engineering Design Optimization, Journal of Mechanical Design 129 (4) (2006) 370–380.

[8] A. Forrester, A. Sobester, A. Keane, Engineering design via surrogate modelling: a practical guide, John Wiley & Sons, 2008.

[9] S. Koziel, L. Leifsson, Surrogate-based modeling and optimization, Springer, 2013.

[10] M. Moustapha, B. Sudret, Surrogate-assisted reliability-based design optimization: a survey and a unified modular framework, Structural and Multidisciplinary Optimization (2019) 1–20.

[11] T. Chatterjee, S. Chakraborty, R. Chowdhury, A critical review of surrogate assisted robust design optimization, Archives of Computational Methods in Engineering 26 (1) (2019) 245–274.

[12] D. R. Jones, M. Schonlau, W. J. Welch, Efficient global optimization of expensive black-box functions, Journal of Global Optimization 13 (4) (1998) 455–492.

[13] Y. Jin, Surrogate-assisted evolutionary computation: Recent advances and future challenges, Swarm and Evolutionary Computation 1 (2) (2011) 61–70.

[14] A. Díaz-Manríquez, G. Toscano, C. A. C. Coello, Comparison of metamodeling techniques in evolutionary algorithms, Soft Computing 21 (19) (2017) 5647–5663.

[15] D. Zhan, H. Xing, Expected improvement for expensive optimization: a review, Journal of Global Optimization 78 (3) (2020) 507–544.

[16] R. K. Tripathy, I. Bilionis, Deep uq: Learning deep neural network surrogate models for high dimensional uncertainty quantification, Journal of Computational Physics 375 (2018) 565–588.

[17] M. Raissi, P. Perdikaris, G. E. Karniadakis, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving non-

linear partial differential equations, Journal of Computational Physics 378 (2019) 686–707.

[18] N.-C. Xiao, M. J. Zuo, C. Zhou, A new adaptive sequential sampling method to construct surrogate models for efficient reliability analysis, Reliability Engineering & System Safety 169 (2018) 330–338.

[19] Z. Xiang, J. Chen, Y. Bao, H. Li, An active learning method combining deep neural network and weighted sampling for structural reliability analysis, Mechanical Systems and Signal Processing 140 (2020) 106684.

[20] Y. Yu, H. Yao, Y. Liu, Structural dynamics simulation using a novel physics-guided machine learning method, Engineering Applications of Artificial Intelligence 96 (2020) 103947.

[21] R. Zhang, Y. Liu, H. Sun, Physics-informed multi-lstm networks for metamodeling of nonlinear structures, Computer Methods in Applied Mechanics and Engineering 369 (2020) 113226.

[22] C. Bernier, J. E. Padgett, Fragility and risk assessment of aboveground storage tanks subjected to concurrent surge, wave, and wind loads, Reliability Engineering & System Safety 191 (2019) 106571.

[23] J. Xin, M. Akiyama, D. M. Frangopol, M. Zhang, J. Pei, J. Zhang, Reliability-based life-cycle cost design of asphalt pavement using artificial neural networks, Structure and Infrastructure Engineering (2020) 1–15.

[24] T. Kim, J. Song, O.-S. Kwon, Probabilistic evaluation of seismic responses using deep learning method, Structural Safety 84 (2020) 101913.

[25] Q. Zou, S. Chen, Resilience-based recovery scheduling of transportation network in mixed traffic environment: A deep-ensemble-assisted active learning approach, Reliability Engineering & System Safety (2021) 107800.

[26] E. Samaniego, C. Anitescu, S. Goswami, V. M. Nguyen-Thanh, H. Guo, K. Hamdia, X. Zhuang, T. Rabczuk, An energy approach to the solution of partial differential equations in computational mechanics via machine learning: Concepts, implementation and applications, Computer Methods in Applied Mechanics and Engineering 362 (2020) 112790.

[27] C. Anitescu, E. Atroshchenko, N. Alajlan, T. Rabczuk, Artificial neural network methods for the solution of second order boundary value problems, Computers, Materials and Continua 59 (1) (2019) 345–359.

[28] B. Sudret, Global sensitivity analysis using polynomial chaos expansions, Reliability Engineering & System Safety 93 (7) (2008) 964–979.

[29] Q. Shao, A. Younes, M. Fahs, T. A. Mara, Bayesian sparse polynomial chaos expansion for global sensitivity analysis, Computer Methods in Applied Mechanics and Engineering 318 (2017) 474–496.

[30] E. Torre, S. Marelli, P. Embrechts, B. Sudret, Data-driven polynomial chaos expansion for machine learning regression, Journal of Computational Physics 388 (2019) 601–623.

[31] B. Bhattacharyya, Structural reliability analysis by a bayesian sparse polynomial chaos expansion, Structural Safety 90 (2021) 102074.

[32] K. Cheng, Z. Lu, Active learning bayesian support vector regression model for global approximation, Information Sciences 544 (2021) 549–563.

[33] K. Cheng, Z. Lu, Adaptive bayesian support vector regression model for structural reliability analysis, Reliability Engineering & System Safety 206 (2021) 107286.

[34] J. Song, P. Wei, M. Valdebenito, M. Beer, Active learning line sampling for rare event analysis, Mechanical Systems and Signal Processing 147 (2021) 107113.

[35] J. Song, P. Wei, M. Valdebenito, M. Beer, Adaptive reliability analysis for rare events evaluation with global imprecise line sampling, Computer Methods in Applied Mechanics and Engineering 372 (2020) 113344.

[36] S. Atkinson, N. Zabaras, Structured bayesian gaussian process latent variable model: Applications to data-driven dimensionality reduction and high-dimensional inversion, Journal of Computational Physics 383 (2019) 166–195.

[37] P. Satria Palar, L. Rizki Zuhal, K. Shimoyama, Gaussian process surrogate model with composite kernel learning for engineering design, AIAA journal 58 (4) (2020) 1864–1880.

[38] Y. Liu, M. Collette, Improving surrogate-assisted variable fidelity multi-objective optimization using a clustering algorithm, Applied Soft Computing 24 (2014) 482–493.

[39] A. I. Khuri, S. Mukhopadhyay, Response surface methodology, Wiley Interdisciplinary Reviews: Computational Statistics 2 (2) (2010) 128–149.

[40] R. H. Myers, D. C. Montgomery, C. M. Anderson-Cook, Response surface methodology: process and product optimization using designed experiments, John Wiley & Sons, 2016.

[41] Y. Wang, Q. Yao, J. T. Kwok, L. M. Ni, Generalizing from a few examples: A survey on few-shot learning, ACM Computing Surveys (CSUR) 53 (3) (2020) 1–34.

[42] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, C. Liu, A survey on deep transfer learning, in: International conference on artificial neural networks, Springer, 2018, pp. 270–279.

[43] T. Hospedales, A. Antoniou, P. Micaelli, A. Storkey, Meta-learning in neural networks: A survey, arXiv preprint arXiv:2004.05439 (2020).

[44] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, Q. He, A comprehensive survey on transfer learning, Proceedings of the IEEE 109 (1) (2020) 43–76.

[45] U. Kirsch, P. Y. Papalambros, Structural reanalysis for topological modifications–a unified approach, Structural and Multidisciplinary Optimization 21 (5) (2001) 333–344.

[46] U. Kirsch, A unified reanalysis approach for structural analysis, design, and optimization, Structural and Multidisciplinary Optimization 25 (2) (2003) 67–85.

[47] U. Kirsch, Reanalysis and sensitivity reanalysis by combined approximations, Structural and Multidisciplinary Optimization 40 (1) (2010) 1–15.

[48] Y. Gao, K. M. Mosalam, Deep transfer learning for image-based structural damage recognition, Computer-Aided Civil and Infrastructure Engineering 33 (9) (2018) 748–768.

[49] S. Dorafshan, R. J. Thomas, M. Maguire, Comparison of deep convolutional neural networks and edge detectors for image-based crack detection in concrete, Construction and Building Materials 186 (2018) 1031–1045.

[50] C. Feng, H. Zhang, S. Wang, Y. Li, H. Wang, F. Yan, Structural damage detection using deep convolutional neural network and transfer learning, KSCE Journal of Civil Engineering 23 (10) (2019) 4493–4502.

[51] L. Bull, P. Gardner, J. Gosliga, T. Rogers, N. Dervilis, E. Cross, E. Papatheou, A. Maguire, C. Campos, K. Worden, Foundations of population-based shm, part i: Homogeneous populations and forms, Mechanical Systems and Signal Processing 148 (2021) 107141.

[52] J. Gosliga, P. Gardner, L. Bull, N. Dervilis, K. Worden, Foundations of population-

based shm, part ii: Heterogeneous populations–graphs, networks, and communities, Mechanical Systems and Signal Processing 148 (2021) 107144.

[53] P. Gardner, L. Bull, J. Gosliga, N. Dervilis, K. Worden, Foundations of population-based shm, part iii: Heterogeneous populations–mapping and transfer, Mechanical Systems and Signal Processing 149 (2021) 107142.

[54] G. Tsialiamanis, C. Mylonas, E. Chatzi, N. Dervilis, D. J. Wagg, K. Worden, Foundations of population-based shm, part iv: The geometry of spaces of structures and their feature spaces, Mechanical Systems and Signal Processing 157 (2021) 107692.

[55] O. Amir, M. P. Bendsøe, O. Sigmund, Approximate reanalysis in topology optimization, International Journal for Numerical Methods in Engineering 78 (12) (2009) 1474–1491.

[56] O. Amir, O. Sigmund, B. S. Lazarov, M. Schevenels, Efficient reanalysis techniques for robust topology optimization, Computer Methods in Applied Mechanics and Engineering 245 (2012) 217–231.

[57] W. Zuo, J. Bai, J. Yu, Sensitivity reanalysis of static displacement using taylor series expansion and combined approximate method, Structural and Multidisciplinary Optimization 53 (5) (2016) 953–959.

[58] W. Zuo, K. Huang, J. Bai, G. Guo, Sensitivity reanalysis of vibration problem using combined approximations method, Structural and Multidisciplinary Optimization 55 (4) (2017) 1399–1405.

[59] L. Bull, P. Gardner, N. Dervilis, E. Papatheou, M. Haywood-Alexander, R. Mills, K. Worden, On the transfer of damage detectors between structures: An experimental case study, Journal of Sound and Vibration (2021) 116072.

[60] M. D. Zeiler, R. Fergus, Visualizing and understanding convolutional networks, in: European conference on computer vision, Springer, 2014, pp. 818–833.

[61] J. Yosinski, J. Clune, Y. Bengio, H. Lipson, How transferable are features in deep neural networks?, arXiv preprint arXiv:1411.1792 (2014).

[62] C. Finn, P. Abbeel, S. Levine, Model-agnostic meta-learning for fast adaptation of deep networks, in: International Conference on Machine Learning, PMLR, 2017, pp. 1126–1135.

[63] X. Lu, F. McKenna, Q. Cheng, Z. Xu, X. Zeng, S. A. Mahin, An open-source framework for regional earthquake loss estimation using the city-scale nonlinear time history analysis, Earthquake Spectra 36 (2) (2020) 806–831.

[64] W. Sun, P. Bocchini, B. D. Davison, Resilience metrics and measurement methods for transportation infrastructure: the state of the art, Sustainable and Resilient Infrastructure 5 (3) (2020) 168–199.

[65] A. Antoniou, H. Edwards, A. Storkey, How to train your maml, arXiv preprint arXiv:1810.09502 (2018).

[66] A. Nichol, J. Achiam, J. Schulman, On first-order meta-learning algorithms, arXiv preprint arXiv:1803.02999 (2018).

[67] A. Rajeswaran, C. Finn, S. Kakade, S. Levine, Meta-learning with implicit gradients (2019) 113–124.

[68] S. Yang, L. Liu, M. Xu, Free lunch for few-shot learning: Distribution calibration, arXiv preprint arXiv:2101.06395 (2021).

[69] S. Bechtle, A. Molchanov, Y. Chebotar, E. Grefenstette, L. Righetti, G. Sukhatme, F. Meier, Meta learning via learned loss, in: 2020 25th International Conference on Pattern Recognition (ICPR), IEEE, 2021, pp. 4161–4168.

[70] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980 (2014).

[71] M. G. Faes, M. Daub, S. Marelli, E. Patelli, M. Beer, Engineering analysis with probability boxes: a review on computational methods, Structural Safety 93 (2021) 102092.

[72] M. Kitahara, S. Bi, M. Broggi, M. Beer, Nonparametric bayesian stochastic model updating with hybrid uncertainties, Mechanical Systems and Signal Processing 163 (2022) 108195.

[73] J. Herman, W. Usher, Salib: an open-source python library for sensitivity analysis, Journal of Open Source Software 2 (9) (2017) 97.

[74] M. Jamil, X.-S. Yang, A literature survey of benchmark functions for global optimisation problems, International Journal of Mathematical Modelling and Numerical Optimisation 4 (2) (2013) 150–194.

[75] C. Dang, P. Wei, J. Song, M. Beer, Estimation of failure probability function under imprecise probabilities by active learning–augmented probabilistic integration, ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part A: Civil Engineering 7 (4) (2021) 04021054.

[76] P. Paris, F. Erdogan, A critical analysis of crack propagation laws, Journal of Fluids Engineering 85 (4) (1963) 528–533.

[77] L. Liu, D. Y. Yang, D. M. Frangopol, Probabilistic cost-benefit analysis for service life extension of ships, Ocean Engineering 201 (2020) 107094.

[78] X. Han, D. Y. Yang, D. M. Frangopol, Probabilistic life-cycle management framework for ship structures subjected to coupled corrosion–fatigue deterioration processes, Journal of Structural Engineering 145 (10) (2019) 04019116.

[79] C. Finn, A. Rajeswaran, S. Kakade, S. Levine, Online meta-learning, in: International Conference on Machine Learning, PMLR, 2019, pp. 1920–1930.

[80] M. Khodak, M.-F. Balcan, A. Talwalkar, Provable guarantees for gradient-based meta-learning, in: International Conference on Machine Learning, PMLR, 2019, pp. 424–433.

[81] Z. Wang, Z. Dai, B. Póczos, J. Carbonell, Characterizing and avoiding negative transfer, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 11293–11302.

[82] N. Tripuraneni, M. I. Jordan, C. Jin, On the theory of transfer learning: The importance of task diversity, arXiv preprint arXiv:2006.11650 (2020).

[83] N. Tripuraneni, C. Jin, M. Jordan, Provable meta-learning of linear representations, in: International Conference on Machine Learning, PMLR, 2021, pp. 10434–10443.

[84] T. Hastie, R. Tibshirani, J. H. Friedman, The elements of statistical learning: Data mining, inference, and prediction, Springer Science & Business Media, 2017.

[85] X. Glorot, A. Bordes, Y. Bengio, Deep sparse rectifier neural networks, in: Proceedings of the fourteenth international conference on artificial intelligence and statistics, JMLR Workshop and Conference Proceedings, 2011, pp. 315–323.

[86] M. Soliman, D. M. Frangopol, A. Mondoro, A probabilistic approach for optimizing inspection, monitoring, and maintenance actions against fatigue of critical ship details, Structural Safety 60 (2016) 91–101.

[87] S. Kim, D. M. Frangopol, Cost-based optimum scheduling of inspection and monitoring for fatigue-sensitive structures under uncertainty, Journal of Structural Engineering 137 (11) (2011) 1319–1331.

[88] T. Moan, E. Ayala-Uraga, Reliability-based assessment of deteriorating ship structures operating in multiple sea loading climates, Reliability Engineering & System Safety 93 (3) (2008) 433–446.

List of Figures

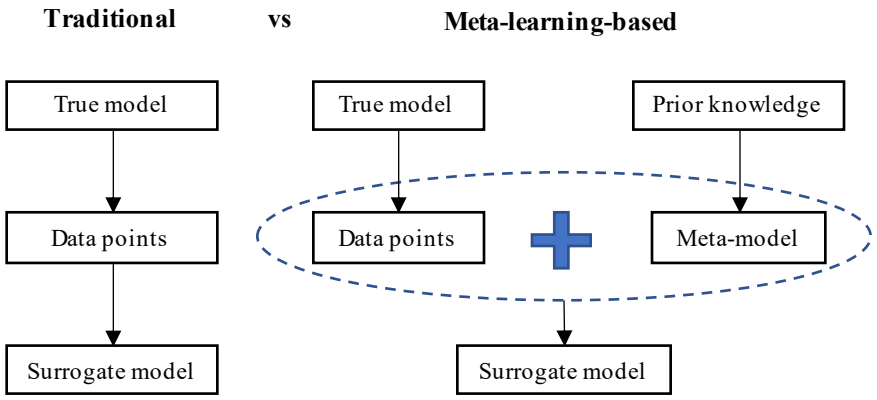


Figure 1: Comparison of traditional surrogate modelling and meta-learning-based surrogate modelling

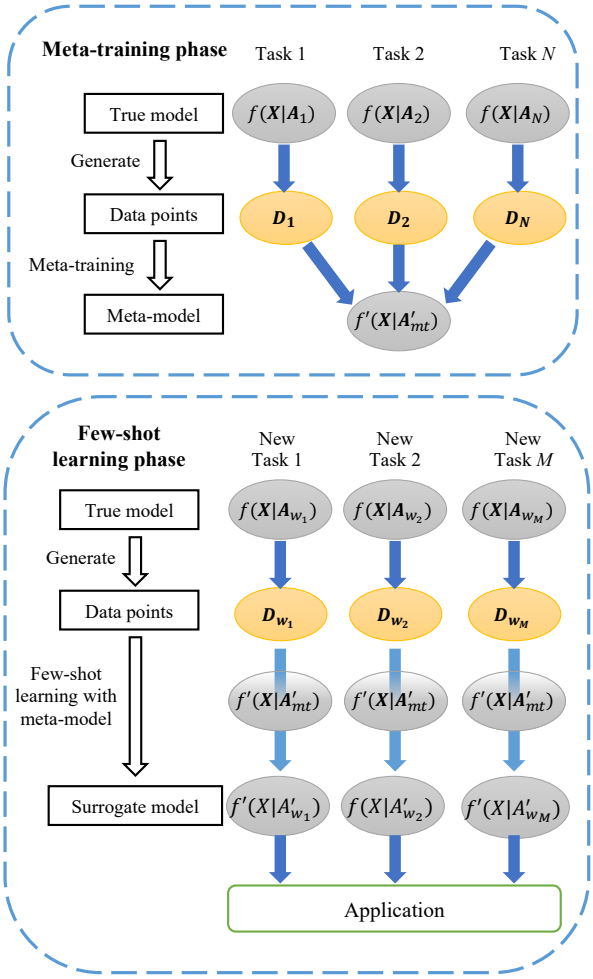


Figure 2: Meta-learning-based surrogate modelling framework

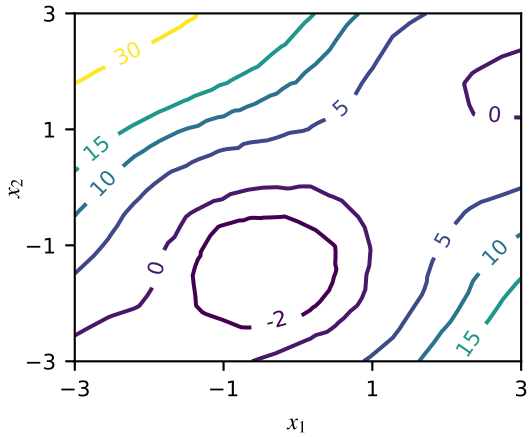
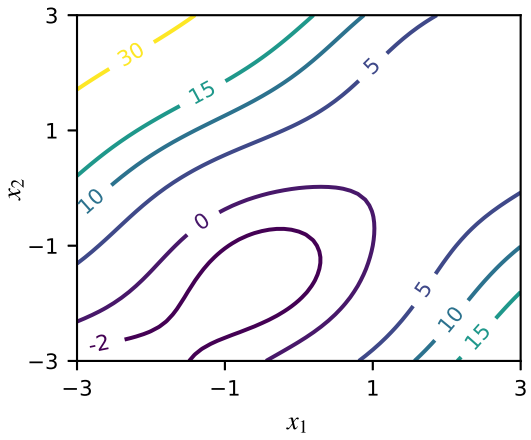
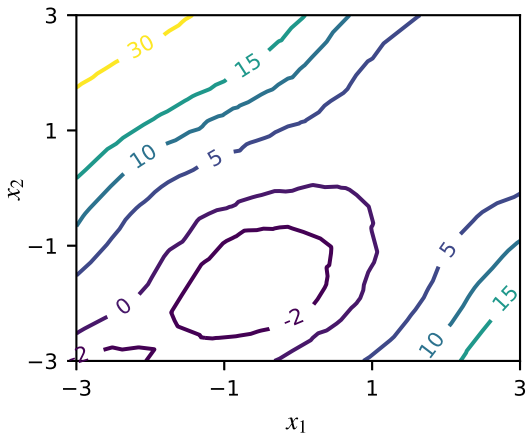


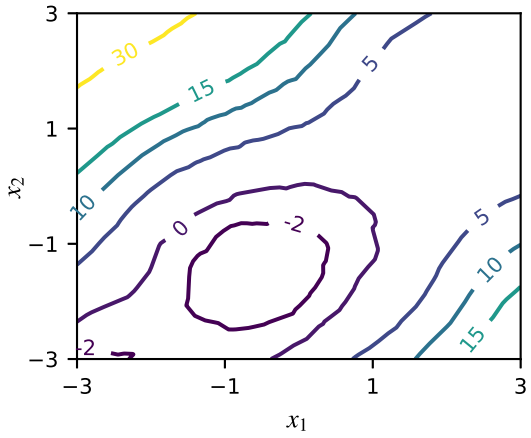
Figure 3: Contour plots of the meta-model of example 1



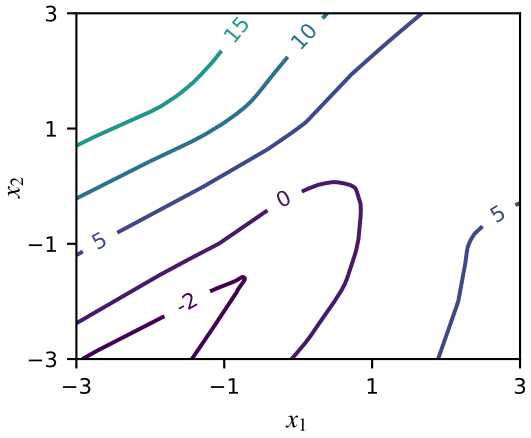
(a) True function



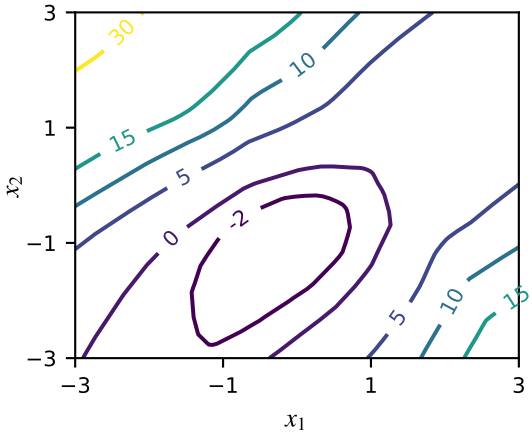
(b) MLSM: K=10



(c) MLSM: K=20

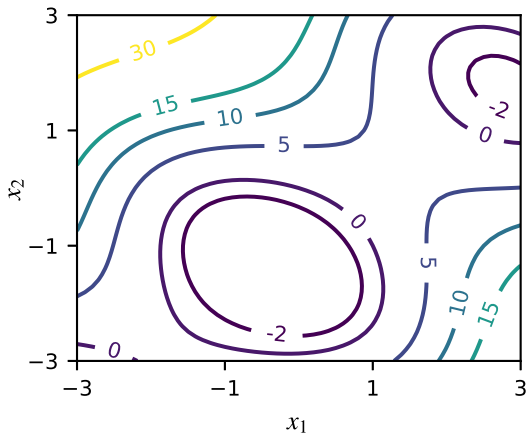


(d) TSM: K=10

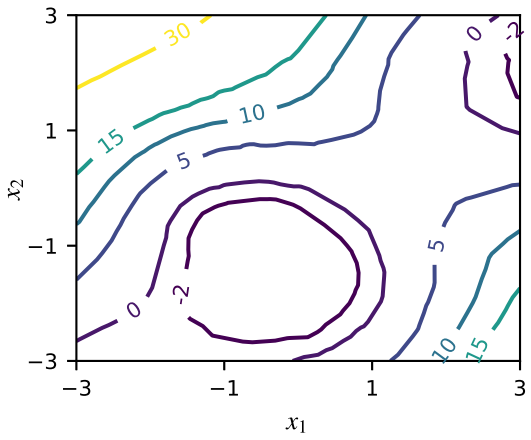


(e) TSM: K=20

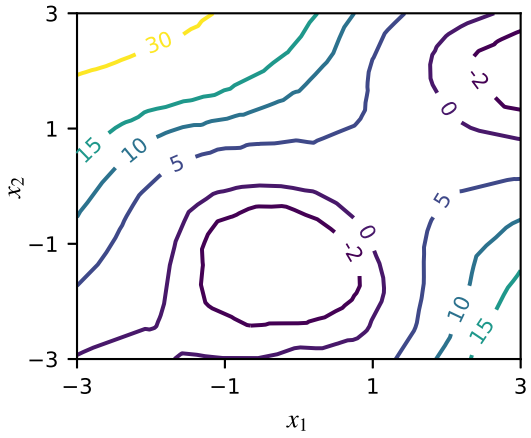
Figure 4: Contour plots of Task I of example 1



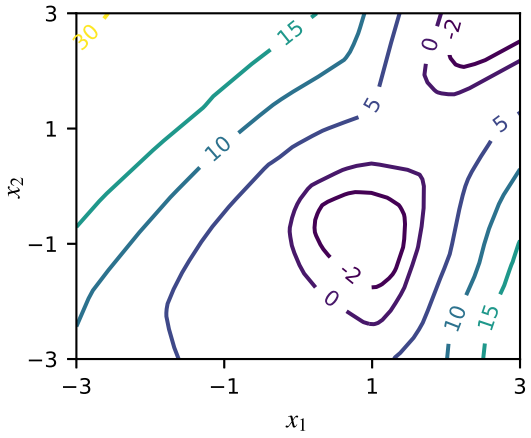
(a) True function



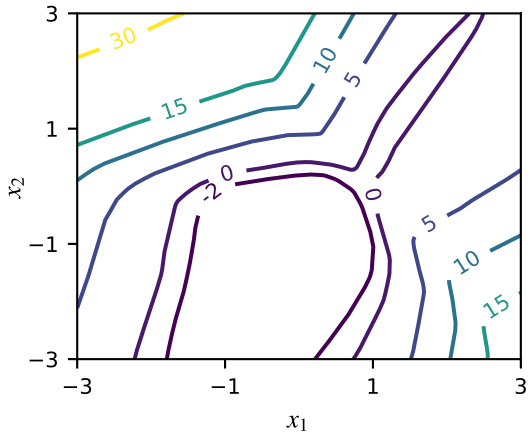
(b) MLSM: K=10



(c) MLSM: K=20

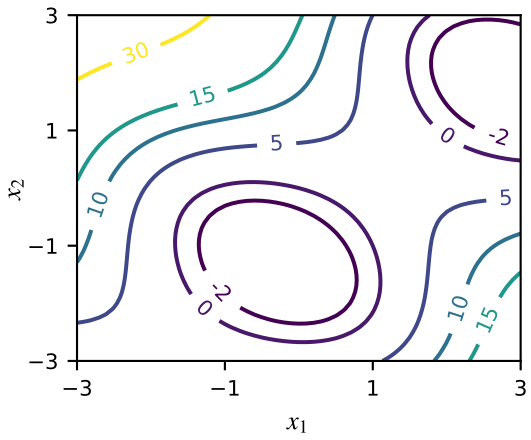


(d) TSM: K=10

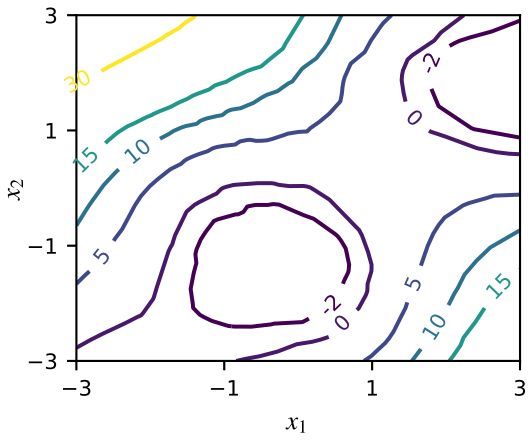


(e) TSM: K=20

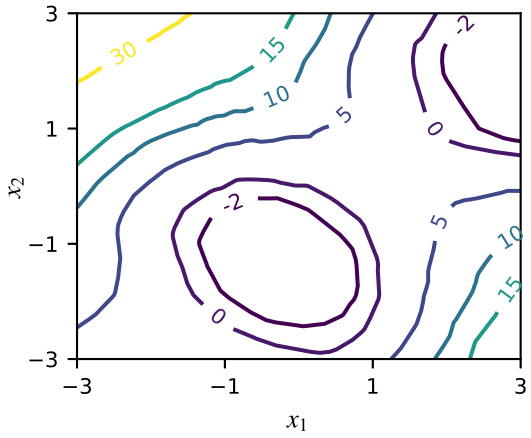
Figure 5: Contour plots of Task II of example 1



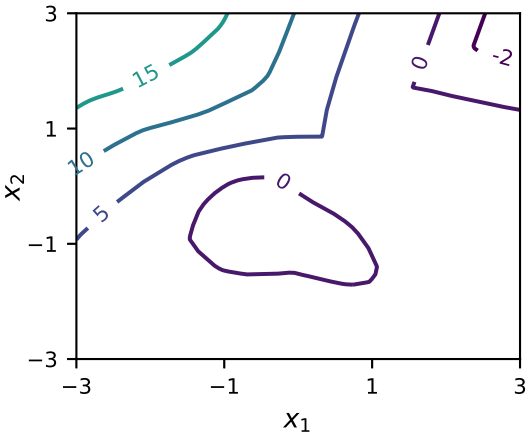
(a) True function



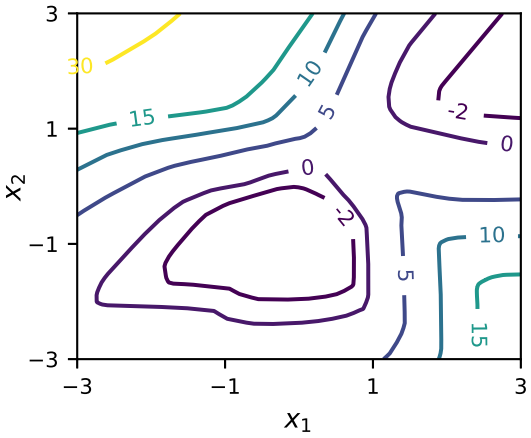
(b) MLSM: K=10



(c) MLSM: K=20



(d) TSM: K=10



(e) TSM: K=20

Figure 6: Contour plots of Task III of example 1

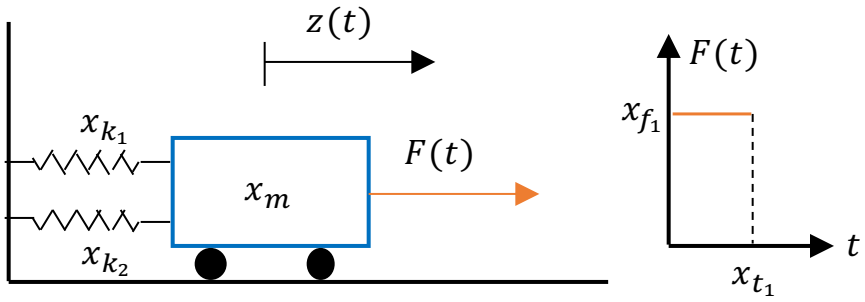


Figure 7: Example 3: nonlinear oscillator [75]

List of Tables

Table 1: Mean squared errors for example 1

Task	MLSM		TSM			
	K=10	K=20	K=10	K=20	K=30	K=40
I	0.14	0.11	12.50	1.73	0.95	0.33
II	1.18	0.46	20.83	4.07	2.60	1.27
III	1.89	0.62	29.87	4.51	2.43	1.19
IV	1.36	0.76	10.72	2.25	0.80	0.43

Note: Each value corresponds to the run with the median value of mean squared errors of 5 runs.

Table 2: Coefficients and mean squared errors for five tasks of example 2

Task	a_1	a_2	Amplitude	MLSM		TSM	
				K=30	K=60	K=120	K=240
I	7	0.1	28.5	5.08	2.30	3.09	1.01
II	21	0.3	81.4	3.79	3.04	28.10	8.55
III	42	0.1	53.7	29.94	12.78	38.71	5.46
IV	21	0.6	139.9	33.64	22.43	77.88	21.54
V	7	0.03	14.8	7.84	3.10	1.08	0.23

Note: Each value is the average of mean squared errors of 5 runs.

Table 3: Sobol's indices for Task I of example 2: a_1=7, a_2=0.1

Index	Meta-model	True model	MLSM	
			K=30	K=60
S_1	0.204	0.308	0.266	0.250
S_2	0.534	0.448	0.419	0.444
S_{13}	0.259	0.251	0.268	0.248
S_{T_1}	0.484	0.560	0.581	0.553
S_{T_2}	0.526	0.439	0.457	0.491
S_{T_3}	0.254	0.243	0.300	0.273

Note: Each value is the average of Sobol's indices of 5 runs.

Table 4: Sobol's indices for Task II of example 2: a_1=21, a_2=0.3

Index	Meta-model	True model	MLSM	
			K=30	K=60
S_1	0.204	0.208	0.209	0.211
S_2	0.534	0.512	0.525	0.516
S_{13}	0.259	0.288	0.251	0.262
S_{T_1}	0.484	0.496	0.481	0.494
S_{T_2}	0.526	0.502	0.526	0.512
S_{T_3}	0.254	0.278	0.256	0.268

Note: Each value is the average of Sobol's indices of 5 runs.

Table 5: Sobol's indices for Task III of example 2: a_1=42, a_2=0.1

Index	Meta-model	True model	MLSM	
			K=30	K=60
S_1	0.204	0.014	0.059	0.035
S_2	0.534	0.970	0.800	0.890
S_{13}	0.259	0.020	0.085	0.044
S_{T_1}	0.484	0.034	0.179	0.102
S_{T_2}	0.526	0.966	0.842	0.915
S_{T_3}	0.254	0.015	0.109	0.053

Note: Each value is the average of Sobol's indices of 5 runs.

Table 6: Sobol's indices for Task IV of example 2: a_1=21, a_2=0.6

Index	Meta-model	True model	MLSM	
			K=30	K=60
S_1	0.204	0.315	0.316	0.335
S_2	0.534	0.221	0.318	0.265
S_{13}	0.259	0.472	0.321	0.362
S_{T_1}	0.484	0.787	0.662	0.732
S_{T_2}	0.526	0.212	0.335	0.280
S_{T_3}	0.254	0.469	0.360	0.404

Note: Each value is the average of Sobol's indices of 5 runs.

Table 7: Sobol's indices for Task V of example 2: a_1=7, a_2=0.03

Index	Meta-model	True model	MLSM	
			K=30	K=60
S_1	0.204	0.157	0.202	0.140
S_2	0.534	0.801	0.484	0.606
S_{13}	0.259	0.047	0.240	0.147
S_{T_1}	0.484	0.205	0.498	0.360
S_{T_2}	0.526	0.796	0.542	0.692
S_{T_3}	0.254	0.040	0.289	0.194

Note: Each value is the average of Sobol's indices of 5 runs.

Table 8: Mean values of coefficients, failure probabilities, and the minimum number of data points required to achieve targeted level of accuracy for example 3

Task	μ_{k_1}	μ_{k_2}	μ_{f_1}	μ_{t_1}	μ_m	μ_r	P_f	MLSM	TSM
I	1.0	0.1	1.1	1.0	1.0	0.7	0.0019	20	100
II	1.0	0.2	1.0	1.0	1.4	0.5	0.0034	70	110
III	0.5	0.1	1.0	1.0	1.0	0.8	0.0072	40	110
IV	1.4	0.3	1.3	0.9	1.0	0.5	0.0129	20	40
V	0.6	0.5	1.0	1.0	1.0	0.5	0.0268	20	70
VI	1.0	0.1	1.0	1.5	2.0	0.5	0.0456	20	40

Table 9: Parameters of the baseline scenario of example 4

Parameter	Mean	COV	Distribution type
Initial crack size a_0 [a]	0.5 mm	0.1	Lognormal
Material parameter C [a]	2.3×10^{-12}	0.3	Lognormal
Material parameter m [a]	3.0	—	Deterministic
Stress range σ_r [a]	20 MPa	0.1	Weibull
Average annual number of cycles [a,b]	5×10^5 cycles/year	0.1	Lognormal
Geometry parameter G [a]	1.12	—	Deterministic
Mean detectable crack size a_μ [c]	1.8 mm	—	Deterministic
Critical crack size for failure a_f [a]	50 mm	—	Deterministic
Cost of inspection C_{ins} [d]	5	—	Deterministic
Cost of repair C_{rep} [d]	50	—	Deterministic
Cost of failure C_f [d]	200	—	Deterministic
Service life t_{end} [d]	25 years	—	Deterministic

[a] From [86]
[b] From [87]
[c] From [88]
[d] Assumed

Table 10: Optimal solutions for testing tasks

Task	Optimal $\mathbb{E}(C_{LC})$	t_1 (year)	t_2 (year)	a_r (mm)
I	22.95	12	17	3.2
II	23.19	12	17	3.3
III	15.59	12	17	3.2
IV	22.27	10	16	2.6
V	89.91	8	16	1.6
VI	15.84	13	18	3.9
VII	19.47	11	16	3.4
VIII	26.86	12	18	3.0

Table 11: Minimum number of data points required to achieve targeted level of accuracy for the true optimal $\mathbb{E}(C_{LC})$

Task	MLSM	TSM
I	80	220
II	100	180
III	40	100
IV	80	160
V	60	220
VI	60	160
VII	80	160
VIII	200	200