Towards a population-informed approach to the definition of data-driven models for structural dynamics

G. Tsialiamanis[1], N. Dervilis, D.J. Wagg, K. Worden

Dynamics Research Group, Department of Mechanical Engineering, University of Sheffield Mappin Street, Sheffield S1 3JD

Abstract

Machine learning has affected the way in which many phenomena for various domains are modelled, one of these domains being that of structural dynamics. However, because machine-learning algorithms are problem-specific, they often fail to perform efficiently in cases of data scarcity. To deal with such issues, combination of physics-based approaches and machine learning algorithms have been developed. Although such methods are effective, they also require the analyser's understanding of the underlying physics of the problem. The current work is aimed at motivating the use of models which learn such relationships from a population of phenomena, whose underlying physics are similar. The development of such models is motivated by the way that physics-based models, and more specifically *finite element* models, work. Such models are considered transferrable, explainable and trustworthy, attributes which are not trivially imposed or achieved for machine-learning models. For this reason, machine-learning approaches are less trusted by industry and often considered more difficult to form *validated* models. To achieve such data-driven models, a *population-based* scheme is followed here and two different machine-learning algorithms from the *meta-learning* domain are used. The two algorithms are the *model-agnostic meta-learning* (MAML) algorithm and the *conditional neural processes* (CNP) model. The two approaches have been developed to perform within a population of tasks and, herein, they are tested on a simulated dataset of a population of structures, with data available from a small subset of the population. Such situations are considered to be similar to having data available from existing structures or structures in a laboratory environment or even from a model and needing to model a new structure with only a few available data samples. The algorithms seem to perform as intended and outperform a traditional machine-learning algorithm at approximating the quantities of interest. Moreover, they exhibit

[1]Corresponding Author: George Tsialiamanis (g.tsialiamanis@sheffield.ac.uk)

behaviour similar to traditional machine learning algorithms (e.g. neural networks or Gaussian processes), concerning their performance as a function of the available structures in the training population, i.e. the more training structures, the better and more robustly the algorithms learn the underlying relationships.

Keywords: Structural dynamics, machine learning, population-based modelling, transfer learning, meta-learning.

---

1. Introduction

Modelling of structural dynamics has been extensively studied throughout the years. The need to study how structures behave and react to the various conditions of their environment is motivated by the need to create safe and long-lasting infrastructure. This need has led researchers to try to understand the physics of various materials and of structural members made from such materials. Such studies led to creation of mathematical models, founded on the understanding of the physics of the various components of the structures.

Numerous methods have been developed to analyse and predict the behaviour of structures, but, arguably, the most common and successful means of structural analysis is the *finite element* (FE) method [1]. The method has been widely successful because of its generality and accuracy, especially for static problems. Although the FE method was developed many years in the past, the advance of computational resources [2] have allowed the modern use of ever more detailed and complicated FE models, maintaining the method as the dominant structural-analysis strategy.

Naturally, models are used to simulate structures before fabrication and after implementation. For the former case, one typically has no available data in order to define the parameters of the model. The selection of the parameters is performed using engineering intuition and available data from experiments or from existing structures, built with similar materials and comprising similar structural members. For the latter case however, the analyst may have acquired some data from the existing structure and may seek to tune the parameters of a model, so that the predictions of the model align with the recorded data. The knowledge acquired from such a model calibration procedure is in the form of values, or distributions of values, for the tunable parameters of the models and can be exploited in similar tasks in the future.

In recent years, because of the vast development of data-driven modelling methods for many disciplines and because of how conveniently such methods fit the framework of using data acquired from a structure in order to model it, data-driven structural modelling has emerged as a viable approach. The

field which has contributed most, is that of *machine learning* [3, 4, 5, 6], and it has been extensively used for the purposes of structural dynamics [7, 8], and quite often for *structural health monitoring* (SHM) [9]. The success of the application of such methods lies in the fact that they learn the underlying relationships of the modelled quantities from data and do not require extensive physical understanding of these relationships by the creator of the model. Data-driven methods are also able to effectively simulate physics of different domains and without the restriction of time and space scales.

However, the ability of data-driven models to learn without knowledge of the underlying physics, comes with a shortcoming - the need for data. Depending on the problem and the algorithm that is used, the amount of available data and their quality might be restrictive in terms of performing appropriate inference. In extreme cases data may not be available at all, making the use of such algorithms difficult or even infeasible. In the field of structural dynamics, this problem is extensive [10]. Data from structures are not always available, need specialised equipment to be acquired and in many cases, the environmental conditions during the acquisition of the data may not be the same as the conditions under which one seeks to model the structure. The latter problem is evident when one performs experiments in a laboratory and uses these data to model a structure in the field.

Physics-based models do not suffer very often from lack of such data. The reduced need for data is balanced by the imposition of physics into the model. Assuming that the physics are accurate to a prescribed level, the model would be expected to be able to generalise well, even if it is calibrated using only a small number of data points [11]. However, the physics are not the only factor that allow these models to generalise that well. An aspect of such models is the convenience of knowledge transfer between members in a population of structures. As mentioned, studying structures started by studying materials and simple components in a laboratory. Subsequently, the acquired knowledge is used to model real and complicated structures based on the behaviour observed in the laboratory. The results of calibrating a model, e.g. a FE model, can provide a prior belief, which could assist in modelling a structure comprising the same materials and same type of members.

Apart from the reduced need for data, a convenient characteristic of physics-based models - such as FE models - is that they are preferred because of their *explainability*. The models have been used for several decades and details about their functionality are known either by analyses that have been performed, or by hands-on experience of the users. This explainability also comes from the fact that their tunable parameters are quite specific and the effect of variations in these parameters can be systematically studied. A large

contrast to the physical trainable parameters is identified if one compares them with the trainable parameters of neural networks, which quite often have no physical meaning and their variations cannot be correlated to specific physically-meaningful effects on the behaviour of the model.

Motivated by this functionality of physics-based models, the current work is focussed on the use of data-driven models, which are informed by populations of structures in order to boost their performance, increase their trustworthiness and impose a sense of explainability. Similar population-based approaches have very recently been followed for SHM problems [12, 13, 14, 15, 16]. The approach proposed here-in is to define data-driven models which are able to adapt to different members of the population, in a similar manner to how corresponding physics-based models are used in a population. In Section 2, the motivation to create such data-driven models is given, as well as the formulation of such models and the potential benefits of using a population-based approach. In Section 3, an introduction to *meta-learning* [17, 18], is given, together with an explanation of its connection with population-based modelling and two algorithms are presented as an attempt to define population-informed models which extract knowledge from the population without the need for the analyser to impose prior knowledge. In Section 4, applications are presented to illustrate the potential of the algorithms in comparison to a traditional machine-learning method, which is trained without exploiting any population knowledge. Finally, in Section 5, conclusions are drawn and future work is discussed.

2. Population-based modelling

A general framework for physics-based modelling follows a mathematical formulation of the physics of some phenomenon and the use of the model to predict responses in hypothetical scenarios. For structural dynamics, the objects under study are structures or structural members and often, the inputs to the models are environmental and operational conditions or loadings imposed by the environment. One example of such a situation is a wind-turbine and the corresponding environmental condition might be the ambient temperature and a potential excitation signal might be the time-history of the wind speed.

The variables affecting the output of the physics-based model can be separated into two categories. The first category comprises the variables $\boldsymbol{x}$ that externally affect the behaviour of the structure. These variables may be environmental variables, such as temperature and humidity or some external loading, such as an excitation force, an earthquake excitation etc. The second category refers to the parameters of the model $\boldsymbol{c}$, which describe quantitatively the physics of the structure which is modelled. Such parameters are often the Young's modulus of the material, the Poisson's ratio etc. The latter

category contains the parameters of a model, which are tuned during the calibration procedure. The quantity of interest $\boldsymbol{y}$ is then given by,

$$\boldsymbol{y}(\boldsymbol{x}) = f^p(\boldsymbol{x}; \boldsymbol{c}) \tag{1}$$

where f^p is a physics-based model and the values of the parameters $\boldsymbol{c}$ are chosen by the framer or the user of the model (at least their nominal values are chosen, since the actual values might vary because of the effect of environmental parameters). Naturally, the above formulation is subjected to uncertainty. The uncertainty is often separated into two categories, the *epistemic uncertainty* and the *aleatory uncertainty*. The first category mainly refers to inconsistencies between the formulation of the model f^p and the phenomenon which it is used to model. The second category refers to inherently random quantities or events. Throughout this paper, vectors are represented by bold symbols, while scalar variables have regular symbols.

A calibration procedure according to observations of the quantities of interest $\boldsymbol{y}$ yields a set of optimal parameters for the values of the tunable parameters $\boldsymbol{c}$ of the model. The convenience of physics-based models lies in the fact that the tunable parameters refer to properties of the materials and the mathematical formulation of the physics is assumed to be common for structures made by similar materials. As a result, to extend the modelling framework in a population scheme, the modelled quantities $\boldsymbol{y}^i$ of the i^{th} structure are given by,

$$\boldsymbol{y}^i(\boldsymbol{x}) = f^p(\boldsymbol{x}; \boldsymbol{c}^i, \mathcal{G}^i) \tag{2}$$

where $\boldsymbol{c}^i$ are the parameters of the i^{th} structure and $\mathcal{G}^i$ is the geometry of the i^{th} structure. The geometry of the structure, which is incorporated in the physics-based model, is another part of the physics that the users incorporate into the model. In the case of FE models, the geometry refers to the number and coordinates of the nodes of the model and the boundary conditions. Transferring knowledge within a population is manual, but formulated and facilitated by the mathematical formulation, as the values of the material parameters $\boldsymbol{c}$ are transferred from one model to the other and the geometry $\mathcal{G}$ is adapted to reflect the new structure, which is to be modelled.

The corresponding framework for structural modelling using data-driven models is similar. A set of quantities of interest $\boldsymbol{y}$ is modelled as a function of some input variables $\boldsymbol{x}$. In the case of modelling a single structure, the data-driven model f^d has trainable parameters $\boldsymbol{\theta}$ and the equation of the predictions is given by,

$$\boldsymbol{y}(\boldsymbol{x}) = f^d(\boldsymbol{x}; \boldsymbol{\theta}) \tag{3}$$

However, in a population-based framework, the approach to transfer knowledge is not the same as in physics-based models. The trainable parameters $\boldsymbol{\theta}$ cannot be trivially transferred or transformed, as in the case of the parameters $\boldsymbol{c}$ and the geometry $\mathcal{G}$, in order to be used for another structure with geometry $\mathcal{G}^*$ within a population.

Under such a framework, attempts at knowledge transfer have been made for SHM in [12, 13, 14, 15]. The first and simplest approach is based on structures in *homogeneous* populations [12]. In that case, the structures are considered similar enough, so that the same model f^d shall perform satisfactorily for all of them. To follow such an approach, one needs to find the set of best fitting parameters $\boldsymbol{\theta}$, by solving the optimisation problem given by,

$$\hat{\boldsymbol{\theta}} = \min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{y}^i(\boldsymbol{x}), f^d(\boldsymbol{x}; \boldsymbol{\theta})) \quad i = 1, 2..., N \tag{4}$$

where $\mathcal{L}$ is some objective function expressing the distance between the predictions of $f^d(\boldsymbol{x}; \boldsymbol{\theta})$ and the observations $\boldsymbol{y}^i(\boldsymbol{x})$ of the i^{th} structure, N is the number of available structures and $\hat{\boldsymbol{\theta}}$ captures the optimal values of the parameters $\boldsymbol{\theta}$, which minimise $\mathcal{L}$. In the equation above, the observations of all N structures are included in the optimisation procedure, making the model common for all the population or, as such models are called in [12], a *form*. Such a model can also be a probabilistic model, yielding a probability density function $p(\hat{\boldsymbol{\theta}})$ instead of single point predictions, i.e. $p(\boldsymbol{y}|\boldsymbol{x}) = f^d(\boldsymbol{x}; \hat{\boldsymbol{\theta}})$.

Structures with significant differences form *heterogeneous* populations, making knowledge transfer more difficult. Various approaches have been followed to deal with such problems. In [14, 19, 20], *domain adaptation* is exploited to perform such knowledge transfer. The problems presented refer to classification of damage on structures with significant differences and how a damage classifier could be transferred and adapted to different structures.

The data-driven methods described so far do not take into account the geometry and do not seek to exploit any other physical knowledge about the structures. To consider the geometry in the inference process, in [13, 15], a graph representation of the structures is used. The representation is performed by breaking down the structures into *irreducible elements* (IEs). Each IE represents a structural element and a graph is constructed by connecting the IEs with each other according to the connectivity of the structural elements. The graphs are attributed according to the properties of the structural members and are used either to classify structures according to their similarities [13, 21] or to perform inference within a population comprising structures created by a set of members of the same type. The geometry is part of the input, as in the case of physics-based models, like FE models, partly compensating for the lack of a physical formulation. The

output of such graph-based models f^g is in this case given by,

$$\boldsymbol{y}^i = f^g(\boldsymbol{x}, \mathcal{G}^i; \boldsymbol{\theta}) \tag{5}$$

where $\boldsymbol{x}$ is some external input which affects the behaviour of the structure and $\mathcal{G}^i$ is the geometry, represented via a graph, of the i^{th} structure.

In order to compare the physics-based and data-driven approaches, one should compare equations (1) and (3). Doing so, a parallelism is observed. In both cases, the modelled quantities $\boldsymbol{y}$ are a function of some inputs-to-the-model $\boldsymbol{x}$ and some parameters of the model, either $\boldsymbol{c}$ and $\mathcal{G}$ in equations (1) and (2) or $\boldsymbol{\theta}$ in equation (3). The major difference between physics-based and data-driven models can be summarised in the difference between the two sets of parameters $\boldsymbol{c}$ and $\boldsymbol{\theta}$.

On the one hand, the parameters $\boldsymbol{c}$ describe the physics of the structure; they vary within an interval of potential values according to the phenomena they describe and are chosen by a user so that they best reflect the physics of the object modelled. Their values are also explainable and *meaningful*; for example, a material with higher value of Young's modulus than some other material is a more stiff material. During a model calibration procedure, a set of model parameters $\boldsymbol{c}$ is sought, which will sufficiently fit the acquired data from some structure. The resulting set of parameters has a strong physical meaning, since the model is bound by the physics imposed by its mathematical formulation. In some cases, calibration may be performed manually using a trial and error technique by an experienced user. Hence, one is confident that, using the same or similar parameters for similar structures, will result in good accuracy for predictions regarding the new structure.

On the other hand, the parameters $\boldsymbol{\theta}$ are selected via the optimisation described by equation (4) and usually have no physical meaning. Taking also into account that *black-box* models are often overparametrised functions (e.g. neural networks); the set of parameters, which best describes a relationship, is not necessarily unique. The latter issue of non-uniqueness creates problems when such models are used in a population-based framework. As in the case of physics-based models, it would be convenient if the population could be described by varying parameters and if smooth alterations of these parameters would correspond to smooth alterations of $\boldsymbol{\theta}$. However, this is not the case, since some parameters $\boldsymbol{\theta}_i$ may describe sufficiently the physics for some structure i, but for a different structure with slightly different characteristics, the optimal set of parameters may not even be in the neighbourhood of $\boldsymbol{\theta}_i$. For the case of physics-based models, the set of all plausible values of $\boldsymbol{c}$ defines all the potential forms of the model and describes a population parametrised by $\boldsymbol{c}$ and its plausible values. In contrast, a data-driven model $f^g(\boldsymbol{x}; \boldsymbol{\theta})$ does not define a meaningful set of mappings for every value of $\boldsymbol{\theta}$,

since for the majority of the values of $\boldsymbol{\theta}$, the model may not resemble any plausible reality.

Trying to create a data-driven model which shall be able to perform within the population and not for a single structure, one could attempt to create a model that takes as inputs the parameters $\boldsymbol{c}$, which describe every structure of the population. The modelled quantities $\boldsymbol{y}$ will then be given by,

$$\boldsymbol{y}(\boldsymbol{x}, \boldsymbol{c}) = f^d(\boldsymbol{x}, \boldsymbol{c}; \boldsymbol{\theta}) \tag{6}$$

This approach outweighs the use of a physics-based model as in equation (1), in the case of existence of epistemic uncertainty for the physics-based model; i.e. when one is not confident that the physical formulation of f^p does sufficiently describe the underlying physics of the problem.

A first problem with this approach is that the available data for the population may often be imbalanced, in the sense that more data may be available for some structure i, and the available data for some other structure j may be restricted. In some cases, this imbalance may not be a major issue, but in general, for structural dynamics, datasets tend to be quite sparse. Commonly, for a small number of structures, plenty of data are recorded, while for other structures only a few samples are available. The imbalance may come from the fact that some structures are in a laboratory environment and tested extensively or accurate models may exist for structures, which are trustworthy and allow testing a structure under various 'what-if' scenarios. Furthermore, if the data of the available structures refer to a small subregion of the parameter space of the structures, the model f^d shall essentially be called on to *extrapolate* for other subregions of parameters, a functionality that most machine-learning algorithms lack. A type of machine learning models that exhibit some extrapolation capabilities are the *physics-informed neural networks* (PINNs) [11].

The second major problem is the *parametrisation* of the structures. For FE models the geometry $\mathcal{G}$ serves as a parametrisation technique; as described by equation (2). For data-driven models and simple cases, such as a population of beams or cantilevers, the parametrisation via the use of the dimensions of the structures is straightforward. However, in other cases, it can be extremely complicated and even infeasible. For example, which parameter vector $\boldsymbol{c}$ should be assigned to the wing of a commercial aircraft and which to the wing of a military aircraft? Both wings are intuitively placed in the same population of structures, but that does not mean that defining a set of parameters which characterise the population is a trivial task. To deal with such issues, as mentioned, in [13], a breakdown of structures into irreducible elements (IEs), and a conversion of structures into graphs is presented. In [15], an attempt is made to use such graphs in order to perform inference

for the population. However, the algorithm presented in [15] requires large amounts of data and an automatic and holistic way to transform structures into graphs. Steps towards an objective transformation of structures into graphs have been recently made in [21].

To deal with the aforementioned problems during the definition of a data-driven problem, a population-based approach is proposed in the current work. The population-based approach, as mentioned, is motivated by PBSHM. The use of a population of structures as a means to train data-driven models is expected to add useful features to these models, that they lack compared to physics-based options. Such features are the ability to be trained and to perform with a few available data samples from a testing structure. Moreover, because the *validation* of models is becoming increasingly important [22], the population-informed approach aims at creating models which, being validated for some structures of the population, are more trustworthy than data-driven models which are not informed somehow by the population. The described approach of using data-driven models may also result in quite explainable models. The explainability is encouraged by the fact that the model shall follow the rules of the population for predictions outside the domain of available data for a testing structure. In contrast completely black-box models may behave almost randomly outside their training domain.

3. Two approaches for population-informed neural network models of structures

3.1. Meta-learning for population-based modelling

The aim of the current work, is to create a data-driven model - more specifically a neural network - which shall be used within a population of structures, in a similar manner that a physics-based model would be used. The similarity which is sought, is that between equations (1) and (3). As mentioned, a physics-based model, following equation (1) with varying parameters $\boldsymbol{c}$, is able to model a potential population of structures which is parametrised by $\boldsymbol{c}$. As a first attempt towards this direction it would be convenient and desirable to have a similar data-driven model as of equation (3), whose parameters $\boldsymbol{\theta}$ would vary and their variation would be in order to model a different member of the population. The space of parameters $\boldsymbol{\theta}$ is essentially $\mathbb{R}^{n_{\boldsymbol{\theta}}}$, where $n_{\boldsymbol{\theta}}$ is the number of trainable parameters of the neural network model f^d. A way to force variation of the parameters to explicitly reflect the various structures of the population is to allow the parameters to vary only within a manifold of $\mathbb{R}^{n_{\boldsymbol{\theta}}}$. Having the parameters "trapped" on this manifold, creates a model f^d whose parameter variations have acquired some physical meaning, i.e. each point of the manifold describes a different structure of the population (which would otherwise be described by $\boldsymbol{c}$).

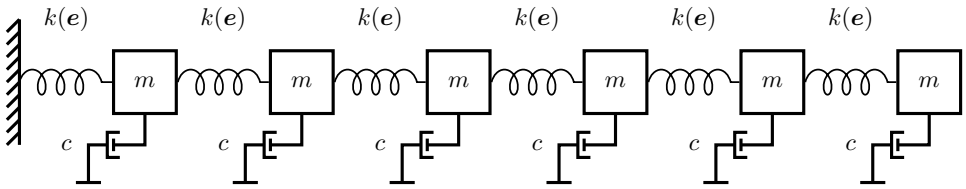


Figure 1: Example of a mass-spring system, with masses m, damping coefficients c and spring stiffness k, which is a function of the environmental conditions $\boldsymbol{e}$.

Regarding the types of populations to which one could apply such a framework, the homogeneity of the population should be taken into account. For quite similar structures, such an approach would be more efficient, similarly to physics-based modelling, where for a homogeneous population, variations of a small subset of parameters (e.g. material-properties parameters) would suffice to model the population. For more heterogeneous populations in the physics-based modelling framework, the geometry of the model might have to change to account for the different structures. The homogeneity of the populations in the current work plays an important role, however, homogeneity in this case should refer to homogeneity of the underlying functions that are modelled. Intuitively, one would expect that homogeneity in the population would lead to homogeneity in the modelled functions. Therefore, it would be safer to follow an approach like the one described in [13] to define a sufficiently-homogeneous population to apply the current methods.

In a structural context, the desired behaviour of the algorithm can be described here via an example. Consider a simple simulated lumped-mass system, as shown in Figure 1. The mass-spring system is defined by its structural parameters: the stiffness k, the damping coefficient c and its mass m. The physics of the system are dictated by the equation of motion,

$$M\ddot{\boldsymbol{y}} + C\dot{\boldsymbol{y}} + K\boldsymbol{y} = \boldsymbol{x}(t) \tag{7}$$

where M, C, K are the mass, damping and stiffness matrices of the system, $\ddot{\boldsymbol{y}}$, $\dot{\boldsymbol{y}}$, $\boldsymbol{y}$ are the vectors of accelerations, velocities and displacements of the degrees of freedom of the system and $\boldsymbol{x}(t)$ is the vector containing the external forces applied to the degrees of freedom of the system as a function of time. Since the system, as mentioned, belongs to a population of similar systems, many such systems are considered here by varying the stiffness parameter k. The system may also be considered to have varying structural parameters as a function of some environmental parameters $\boldsymbol{e}$; e.g. temperature, humidity etc. In the current case, for simplicity, only the stiffness is considered to be affected by the environmental conditions.

For a single structure, a neural network, can be trained to predict some features of the system; e.g. its natural frequencies, a *frequency response*

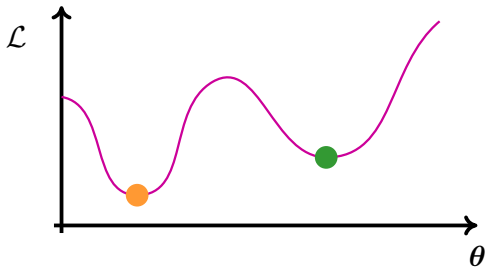


Figure 2: Schematic representation of an objective function $\mathcal{L}$ as a function of the trainable parameter θ and two local minima shown in orange and green.

function (FRF), a *transmissibility* etc. This is a single task τ_i and can be effectively dealt with (and has been performed in the past), given enough training data denoted as,

$$\mathcal{D}_i = \{(\boldsymbol{e}_j, \boldsymbol{y}_j)\}, \quad i = 1, 2...n, \quad j = 1, 2...m \tag{8}$$

where $\boldsymbol{e}_j$ is the j^{th} instance of environmental conditions and $\boldsymbol{y}_j$ the corresponding vector of quantities that are being modelled. To illustrate the desired form of data-driven models, in Figure 2, the value of the loss function $\mathcal{L}$ is shown (the magenta curve) as a function of the trainable parameters $\boldsymbol{\theta}$ (the parameter space will actually be multidimensional with very high dimensionality for neural networks, but for the sake of simplicity and visualisation it is depicted as one-dimensional). In the same figure, two local minima are also depicted; whether they are local or global minima is not of interest for the current case study, as long as the accuracy of the model for the specific set of parameters is acceptable. Consider the values of the loss function, as well as of the minima, to correspond to the whole range of input values: i.e. the value of the loss $\mathcal{L}$ is the *total discrepancy* between all the potential predictions of the model and the true underlying relationship of the environmental conditions $\boldsymbol{e}$ and the quantity $\boldsymbol{y}$ predicted.

The orange point corresponds to a lower loss value, meaning that the corresponding parameter values are the optimal choice for the current task. The green point is also a local minimum and its accuracy may also be acceptable for the purposes of the task. Considering the system within a population-based scheme, another structure is introduced, which is to be modelled using the same model (i.e. the same architecture neural network), and the same loss function. The newly-introduced loss function landscape is shown in Figure 3 (blue curve). As shown, the parameters which corresponded to the minimum for the previous structure, do not correspond to a local minimum. However, one of the two minima may have moved to a neighbouring point, the light green point in the same figure.

This example is case specific, but aims at demonstrating a problem that may

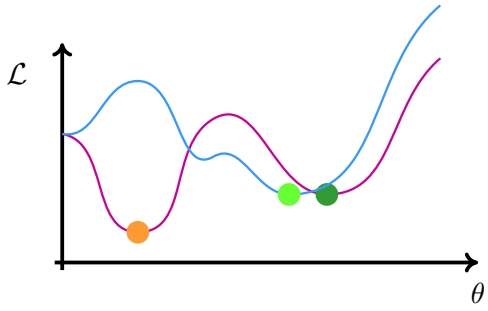


Figure 3: Objective function $\mathcal{L}$ for two tasks (magenta and blue lines) as a function of the trainable parameter Θ and three local minima shown in orange, green and light green.

often occur when training data-driven models in a population-based scheme. When considering a new task/structure, the minima of the first task most probably will not be minima for the new task. However, some minima may transition smoothly to neighbouring points of the parameter space, as the underlying parameters that describe the tasks vary. Therefore, a family of models may describe a variety of tasks, with their trainable parameters $\boldsymbol{\theta}$ lying on a specific area/manifold of the parameter space. As the tasks vary, so shall the parameters that correspond to the local minimum, but always on the same manifold/area of the parameter space.

To further visualise the idea above, a *fibre bundle* [23, 24] is considered. The base manifold of the fibre bundle is the manifold of the tasks $\mathcal{T}$. On this manifold various tasks τ_i are located. The fibre corresponding to each point τ_i of the base manifold, is the manifold shaped by the loss function $\mathcal{L}$ on the parameter space Θ. Consider the set of points of the fibre bundle, which is the collection of local minima for each task. Among these points, the desired models lie on a *cross section*, i.e. a collection of points, across the whole bundle, which are glued together. Given such a cross section, when one needs to move to a neighbouring task, a local minimum for the new task shall exist within the neighbourhood of the current task. The idea is depicted in Figure 4. In the figure, some tasks τ_i are shown with their corresponding fibres (magenta vertical lines), defined as the corresponding losses in the parameter space. The green- and orange-coloured points represent local minima on these functions. As shown, for the orange local minima, as one moves towards neighbouring tasks, the minimum remains glued to the minimum for a small area of the task space, but after some point, it is not a minimum anymore. In contrast, for the green minima, one can navigate through the whole manifold $\mathcal{T}$ and for each step in the neighbour of the current task, a local minimum will also exist in the neighbourhood of the current local minimum.

Fibre bundle shaped using as base manifold the manifold $\mathcal{T}$ of tasks τ_i, each fibre corresponds to the loss function manifold $\mathcal{L}_i$ in the parameter space θ.

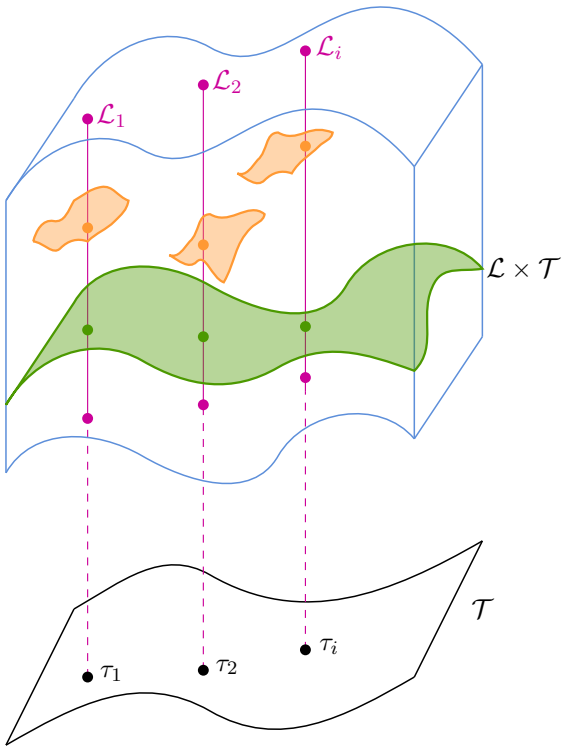


Figure 4: Schematic of a fibre bundle of the loss functions of a data-driven model across a population of tasks. The base manifold $\mathcal{T}$ is formed by points representing the tasks τ_i of the population. To every task τ_i, a fibre $\mathcal{L}_i$ corresponds, which represents the values of the loss function L for different values of the trainable parameters of the data-driven model. The orange and green points represent parts of the fibre bundle where local or global minima are observed regarding the loss function. The fibres intercepting an orange area have a local or global minimum at the intersection point between the fibre and the orange area. The same applies for the green area. However, the green area is a continuous cross section along the whole bundle.

Also, local minima are shown with coloured points (green and orange).

From a physical point of view, the idea above coincides with the idea of forcing the model to approximate the underlying physics of the population of systems. A neural network may be trained to have acceptable accuracy for the task in hand, but this does not mean that it has learnt the underlying physics of the problem. Such generalisation problems may occur when the training data come from specific operational conditions but the testing data come from different operational conditions. According to the idea presented, one can consider that the points on the green cross-section in Figure 4 correspond to a family of models that have learnt the underlying physics of the population. The orange points cannot be considered appropriate captures of the underlying physics, because as the parameters of the systems change smoothly, the values of the trainable parameters of the model, which

correspond to a local minimum, should also change smoothly; however, these points do not have this property.

Although the existence of such a cross section in the fibre bundle is not guaranteed, it might exist for a population of structures with common underlying physics. Consider the case of building a neural network model which takes into account the parametrisation of the structures and aims at approximating the behaviour of the structures of the population - similar to the model f^d of equation (6). In that case, f^d approximates some quantities of interest $\boldsymbol{y}$ as,

$$\boldsymbol{y} = f^d(\boldsymbol{x}^e, \boldsymbol{x}^\tau) \tag{9}$$

where $\boldsymbol{x}^e$ are the inputs describing the external/environmental parameters and $\boldsymbol{x}^\tau$ are the input variables describing the structure/task. Let f^d be a feedforward neural network with specified architecture and, without loss of generality, consider it to be the neural network of Figure 5. Consider a second family of neural networks $\bar{f}^d$ of the same architecture of hidden layers and output layer but with input variables only the environmental variables $\boldsymbol{x}^e$, defined as,

$$\boldsymbol{y}(\tau) = \bar{f}^d(\boldsymbol{x}^e) \tag{10}$$

where the output is a function of the task τ. Without loss of generality, let the $\bar{f}^d$ neural networks be like the one shown in Figure 6. The neural networks f^d and $\bar{f}^d$ can be equivalent for each task τ and the corresponding inputs $\boldsymbol{x}^\tau$ by setting their common trainable parameters to be equal, except for a small subset of them. The common trainable parameters of the two neural networks of Figures 5 and 6 are the weights of the grey connections and the biases of their neurons. For some task τ_i, the input values $\boldsymbol{x}^\tau$ for the first neural network are constant. Therefore, in the second neural network, for the same task τ_i, by varying some of its trainable parameters to account for the missing contribution of $\boldsymbol{x}^\tau$, one can get a neural network equivalent to f^d for constant $\boldsymbol{x}^\tau$ and varying $\boldsymbol{x}^e$; a trivial solution is to vary the bias terms of the hidden layer to compensate for the effect of the missing neural-network connections of $\boldsymbol{x}^\tau$, the red connections in Figure 5. Denoting the trainable parameters of $\bar{f}^d$, which need to vary to capture the effects of varying $\boldsymbol{x}^\tau$, as $\boldsymbol{\theta_v}$ yields,

$$f^d(\boldsymbol{x}^e, \boldsymbol{x}^\tau) = \bar{f}^d(\boldsymbol{x}^e, \boldsymbol{\theta_v}) \tag{11}$$

Computing the derivatives with respect to $\boldsymbol{x}^\tau$, and taking into account the chain rule on the right hand side, yields,

$$\frac{\partial f^d}{\partial \boldsymbol{x}^\tau} = \frac{\partial \bar{f}^d}{\partial \boldsymbol{\theta}_v} \frac{\partial \boldsymbol{\theta}_v}{\partial \boldsymbol{x}^\tau} \tag{12}$$

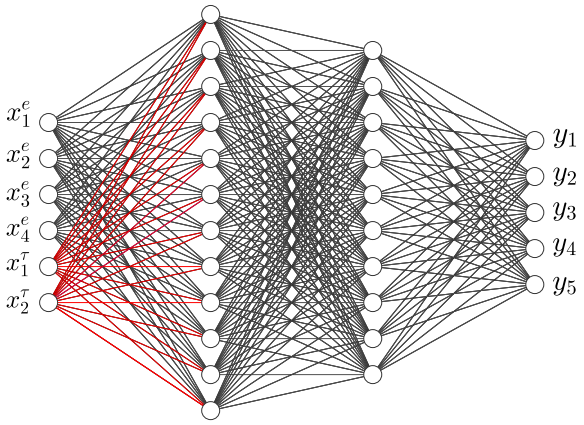


Figure 5: Neural network modelling the population with inputs regarding the external variables $\boldsymbol{x}^e$ and the structure $\boldsymbol{x}^\tau$.

The quantities $\frac{\partial f^d}{\partial \boldsymbol{x}^\tau}$ and $\frac{\partial \bar{f}^d}{\partial \boldsymbol{\theta}_v}$ are continuous for continuous activation functions (or piece-wise continuous for activation functions such as relu or leaky-relu), since they are derivatives of the output of neural networks with respect to some of the input variables or some of their trainable parameters. As a result $\frac{\partial \boldsymbol{\theta}_v}{\partial \boldsymbol{x}^\tau}$ should also be continuous, except for the case that $\frac{\partial \bar{f}^d}{\partial \boldsymbol{\theta}_v}$ is equal to zero. However, the derivative of the output of a neural network with respect to its trainable parameters is rarely exactly equal to zero, given that the parameters $\boldsymbol{\theta}_v$ emulate the effect of the task-describing variables $\boldsymbol{x}^\tau$ they are not expected to be zero, since the task affects largely the quantities $\boldsymbol{y}$. As a result $\frac{\partial \boldsymbol{\theta}_v}{\partial \boldsymbol{x}^\tau}$ is indeed continuous, meaning that, as the parameters $\boldsymbol{x}^\tau$ that characterise the task τ vary smoothly, so do the trainable parameters $\boldsymbol{\theta}_v$ of the $\bar{f}^d$ neural network, for which $\bar{f}^d$ is equivalent to the f^d neural network, which approximates adequately the underlying physics of the population. Although the above study is not a proof, it indicates that, if a latent set of continuous variables which describe the structures exists, the model f^d can accommodate the changes in the behaviour by smoothly altering the values of the trainable parameters. Thus, the cross section of the fibre bundle of Figure 4, which connects the values of the trainable parameters for which error minima are achieved, may exist for a population of heterogeneous structures.

3.2. Model-agnostic meta-learning

For reasons covered in previous sections, data-driven models often lack the ability to generalise well; this is often caused by the lack of sufficient data for proper training or because one attempts to use such models on data outside their training domain. To deal with such issues, several techniques have been used. Such techniques are often referred to as *few-shot learning* [25]. Methods belonging to the few-shot learning domain are: the Siamese

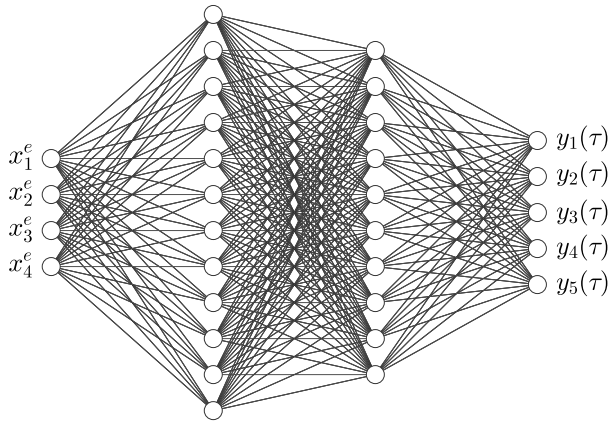


Figure 6: Neural network modelling the population with inputs regarding only the external variables $\boldsymbol{x}^e$.

neural networks [26], the triplet network [27], the matching network [28] etc. Another way of dealing with such lack-of-data problems is *transfer learning*, which attempts to solve these problems by exploiting knowledge about models trained on some domain by transferring it to another domain. This knowledge is usually in the form of some feature extractor [29], or by seeking a domain, onto which data from different tasks can be mapped and a common model can be used [19, 30].

Another approach that aims at dealing with the problem of lack of data is *meta-learning* [18]. Meta-learning can be considered as the domain of machine learning that aims at developing algorithms that 'learn to learn'. A general idea behind such an approach to machine learning, is that the optimisation techniques that are being used - such as gradient descent [3, 4] or Adam [31] - are general purpose algorithms and therefore, not targeted to the problem that one seeks to solve. Furthermore, in the case of population-based modelling, such optimisation algorithms do not exploit knowledge across the population to enhance their performance. Consequently, the goal of meta-learning is to force the training procedure to be effective for a family of tasks; a family of tasks can be a collection of similar problems, domains, or, as in the current work, a population of phenomena with similar underlying physics. A physics-based version of this procedure is described in [16], where the underlying physics of the modelled phenomenon are defined via parametric functions. In the aforementioned work, the members of the population behave according to the pre-defined functions, but also according to tunable parameters, which are tuned in a Bayesian manner when data are acquired from a structure. The definition of the underlying relationship is done by the analyser, reducing the need for data to fit a model for a new structure with a few available data samples. In the current work, the proposed methods are expected to learn an underlying relationship from a

population of structures via meta-learning.

A formalisation of meta-learning is given in [32], where its objective function is defined as the maximisation of,

$$\mathcal{L} = \mathcal{I}(\mathcal{T}, D^v) - \beta\mathcal{I}(\mathcal{T}, D^t) \tag{13}$$

where $\mathcal{T}$ represents the different tasks (denoted as Z in the original work), D^v and D^t represent the unseen data and the training data correspondingly, β is a regularisation parameter and $\mathcal{I}$ is the mutual information. Maximisation of the first term of the specific objective function means maximisation of the information provided by the task τ_i regarding the relationship of the unseen data D^v, i.e. the model should be able to make predictions for the testing data more efficiently when the task is given or inferred. The second term, as stated in [32] is a regularisation term which aims at reducing the mutual information between the training data and the task in order to extract more informative features for the unseen data. The regularisation term aims at discouraging the model from overfitting to the training data and to learn task-independent features. Optimising a population-informed model using the above equation

Such a scheme matches the framework proposed in the current work. In the case of having a physics-based model which describes appropriately the physics of the population, as in [16], a small number of observations from a new structure should suffice to define the parameters of the model and to provide accurate predictions. Similarly, in the current work, a family of neural network models is sought, which matches the underlying physics of a population and, with only a small number of available training data, shall be able to make accurate predictions. Therefore, considering that the current task τ_i is described by the available training data, the maximisation of the mutual information between the available data and the unseen data, could provide such a family of neural-network models.

Meta-learning shows a similar approach to the problem of lack of data as *multi-task learning* [33]. Methods of the multi-task learning domain involve parameter sharing between the initial layers of neural networks. Examples of parameter sharing be can found in [34] and [35], where hard and soft parameter sharing techniques are presented respectively. Meta-learning algorithms could be considered to be a subset of multi-task learning. The main difference between the two is that meta-learning aims at developing algorithms which are able to learn with a few data samples, in contrast to multi-task learning, where often the size of the trainable parameters of a model are reduced by using heuristics, such as whole neural-network-layer transferring. Another difference is that many meta-learning algorithms are focussed on the learning procedure itself, i.e. the optimisation of the model

parameters.

Several methods are available to perform meta-learning as discussed in [18]. In the current work, the method selected to perform meta-learning within a population of structures is *model-agnostic meta-learning* (MAML) [36]. The method aims at training a model, which can quickly - i.e. with a few training steps and with a few samples - adapt to a newly-presented task τ, which comes from a family of tasks $\mathcal{T}$.

In the original work [36], MAML is exploited to enhance the performance of a classifier regarding the Omniglot dataset [37]. The dataset comprises 1623 handwritten characters from 50 different alphabets, written by 20 different people. The resulting dataset has many different characters, but not enough to train a convolutional neural network classifier to perform sufficiently in recognising from which alphabet the characters come.

Altering the traditional machine learning framework and creating various tasks, referred to as K-shot N-way classification, a different approach to the problem is presented. The new goal is to define classifiers which distinguish between N classes of the images having K samples from each class, where K is often a small number and N is smaller than the number of all available classes. Considering each K-shot N-way problem of the aforementioned dataset as a different task, one should seek a way to exploit knowledge from the whole population of tasks, in order to create models that are able to accurately classify the data into N classes, even if the number K is extremely small; in some cases K may even be unity.

The algorithm in the case of classification yields quite satisfactory results; it is also tested in a regression problem and reveals promising results. In [36] and [38] the method is used to approximate relationships which belong to a parameteric family, transferring knowledge from a set of available observations for some members of the population. In the current work, the underlying physics of the population are defined by the parameteric functions, whose exact relationship is considered to be unknown during inference. As in other works, the algorithm does not need any prior knowledge of the potential values of the parameters of the underlying relationships.

Following [36], the algorithm here considers a model $f_{\boldsymbol{\theta}}$ with trainable parameters $\boldsymbol{\theta}$. In the original work, as well as in the current work, the model $f_{\boldsymbol{\theta}}$ is a neural network. The common way of training a neural network for a supervised-learning scheme, is to acquire a set of data $\mathcal{D}^{\tau_i}$ for a single task τ_i and perform backpropagation steps in order to find the set of parameters $\boldsymbol{\theta}^*$, which minimises a loss function $\mathcal{L}^{\tau_i}(f_{\boldsymbol{\theta}})$, summarised by,

$$\boldsymbol{\theta}^* = \min_{\boldsymbol{\theta}} \mathcal{L}^{\tau_i}(f_{\boldsymbol{\theta}}; \mathcal{D}^{\tau_i}) \tag{14}$$

where the dataset $\mathcal{D}^{\tau_i}$ is of the form $\{(\boldsymbol{x}_1, \boldsymbol{y}_1), (\boldsymbol{x}_2, \boldsymbol{y}_2)...(\boldsymbol{x}_N, \boldsymbol{y}_N)\}$ and $(\boldsymbol{x}_j, \boldsymbol{y}_j)$ is the j^{th} pair of inputs $\boldsymbol{x}_j$ and target output values $\boldsymbol{y}_j$. The model with parameters $\boldsymbol{\theta}^*$ is considered *trained* and can be used to predict values of the unseen quantities $\boldsymbol{y}^*$ for new values of the inputs $\boldsymbol{x}^*$.

The MAML approach aims at defining a model, as the initialisation point of neural networks, which shall adapt quickly to new tasks. The difference to traditional learning is that MAML encourages the model to find a point in the trainable parameter space of the neural network, from where, with a few backpropagation steps, the model should be able to converge to a task-specific set of trainable parameters quite quickly. This point is found by performing two sets of backpropagation steps. Initially backpropagation steps are performed for a set of tasks $\mathcal{T}_t \subseteq \mathcal{T}$. The updated parameters $\boldsymbol{\theta}'$, as a result of this first backpropagation step, the *inner update*, are given by,

$$\boldsymbol{\theta}' = \boldsymbol{\theta} - \alpha \nabla \mathcal{L}(f_{\boldsymbol{\theta}}; \mathcal{D}^{tr}_{\tau_i}) \tag{15}$$

where α is the value of the inner updates' learning rate, $\mathcal{L}$ is a loss function, $\nabla \mathcal{L} = [\frac{\partial \mathcal{L}}{\partial \theta_1}, \frac{\partial \mathcal{L}}{\partial \theta_2}, ... \frac{\partial \mathcal{L}}{\partial \theta_i} ...]^T$ and $\mathcal{D}^{tr}_{\tau_i}$ is a dataset sampled from the task τ_i. After performing updates for tasks $\tau_i \in \mathcal{T}_t$, the second set of updates, the *meta-updates* are performed according to the loss function defined as,

$$\min_{\boldsymbol{\theta}} \sum_{\tau_i \in \mathcal{T}} \mathcal{L}_{\tau_i}(f_{\boldsymbol{\theta}'}) \tag{16}$$

so the values of the trainable parameters θ are finally updated as,

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \beta \nabla_{\boldsymbol{\theta}} \sum_{\tau_i \in \mathcal{T}_t} \mathcal{L}(f_{\boldsymbol{\theta}'_i}; \mathcal{D}^{m}_{\tau_i}) \tag{17}$$

where β is the value of the meta-updates' learning rate and $\mathcal{D}^{m}_{\tau_i}$ are the data sampled from task τ_i for the meta-update. An important aspect of the equation above is that the gradients are calculated not with respect to the updated values $\boldsymbol{\theta}_i$, as it would be expected, but with respect to the values of the model parameters $\boldsymbol{\theta}$ before the inner update. The procedure aims at training towards a point $\boldsymbol{\theta}^*$ from where task-specific model-parameter updates shall result in minimisation of the loss function. By updating the model parameters using gradients with respect to the parameters $\boldsymbol{\theta}$ instead of $\boldsymbol{\theta}'_i$, the error is backpropagated through the inner updates as well. As a result, information is drawn from the inner updates, making the resulting set of parameters $\boldsymbol{\theta}^*$ not an optimal set of parameters for all the tasks, but a set of parameters from where training updates, similar to the inner-loop updates, will result in a task-specific optimised model. The procedure is described in Algorithm 1. Note that compared to the equations of the original work [36], the notation $p(\mathcal{T})$ is not used, since the concept of a distribution of tasks

may be confusing.

The goal of such an approach is to find an initialisation point in the parameter space for the neural network, which is "sensitive" to the change of task. Given such a point, one can make the model adapt quite quickly to data from a new task. At the same time, it is expected that since the model is sensitive to the change of task, it shall also be able to adapt with a few training samples, given that these training samples sufficiently characterise the task. Backpropagation derives a limited amount of information from every available data sample; therefore, following the MAML approach, one would expect to balance the lack of information, from which a small dataset $\mathcal{D}_{\tau_t}$ for some task τ_t suffers, with the information encoded from the population while training the MAML model.

The reason that such a training procedure assists in training models within a population-based scheme might be explained by the no free-lunch theorem. As described in [39], considering a hypothesis space $\mathcal{H}$ as the total space of functions where one shall seek a solution to the a learning task, the error $\mathcal{R}(h)$ of a function $h \in \mathcal{H}$ is bounded according to,

$$\mathcal{R}(h) \leq \inf_{h \in \mathcal{H}} \mathcal{R}(h) + \epsilon_{opt} + \epsilon_{stat} + \epsilon_{appr} \tag{18}$$

where ϵ_{opt} is the error of the optimisation procedure and refers to the learning algorithm, ϵ_{stat} is the statistical error referring to the random selection of training data and ϵ_{appr} is the approximation error, which refers to the error regarding the subspace $\mathcal{H}_\delta \subseteq \mathcal{H}$ where the search is performed; for example because of the random initialisation of the trainable parameters the search might be restricted to a small subspace of the whole space of functions. The term $\inf_{f \in \mathcal{H}} \mathcal{R}(h)$ refers to the approximation capabilities of the selected model, which in the case of neural networks should be minimal, since they are universal approximators. Meta-learning algorithms such as MAML, might be a proper way of training models in a population-based framework, since they aim at reducing the total error by reducing both the approximation error and the optimisation error. The approximation error might be reduced, since the algorithm picks an initialisation point from where solutions to the tasks of the studied task family should be close. Thus the search space $\mathcal{F}_\delta$ is implicitly reduced, making the search easier. The optimisation error might be reduced because the outer update of the algorithm backpropagates the error through the inner optimisation step; hence, the initialisation point is not only close to a local minimum, but also only a few backpropagation steps away.

3.3. A task identification subnetwork for population-based modelling

A way in the literature to infuse population knowledge into a learning algorithm is by using Gaussian processes and defining a proper mean and

Algorithm 1 Model-agnostic meta-learning (MAML)

Require: Family of training tasks $\mathcal{T}_{tr}$, testing tasks $\mathcal{T}_t$ loss function $\mathcal{L}$, α, β learning rates, model parameters $\boldsymbol{\theta}$
for each training epoch **do**
  for each task $\tau_i \in \mathcal{T}_{tr}$ **do**
    Sample a set of data $\mathcal{D}_{\tau_i}^{tr}$ for training of the current task τ_i
    $\boldsymbol{\theta}_i' \leftarrow \boldsymbol{\theta} - \alpha \nabla \mathcal{L}(f_{\boldsymbol{\theta}}; \mathcal{D}_{\tau_i}^{tr})$
  end for
  Sample data for meta-training update from every task $\tau_i \in \mathcal{T}_{tr}$ and form the
  meta-training dataset $\mathcal{D}_{\tau_i}^{m}$
  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \beta \nabla_{\boldsymbol{\theta}} \sum_{\tau_i \in \mathcal{T}_t} \mathcal{L}(f_{\boldsymbol{\theta}_i'}; \mathcal{D}_{\tau_i}^{m})$
end for
Testing time
Sample a task of interest τ_j from $\mathcal{T}_t$ with available data $\mathcal{D}_{\tau_j}^{t}$
for each task-specific training step **do**
  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}^* - \alpha \nabla \mathcal{L}(f_{\boldsymbol{\theta}}; \mathcal{D}_{\tau_j}^{t})$
end for

covariance matrix [7, 40]. Such an approach yields quite good results, but a requirement of the method is that the mean or covariance functions can be defined by the analyser, who is imposing knowledge into the algorithm. This might not be a simple task and in some cases it might be a very difficult one; for example, in the case of multidimensional output.

Motivated by the formalisation of meta-learning, as it is given by equation (13), and by the way that Gaussian processes [41] function, an assembly of two models for population-based modelling can be considered as an attempt to define a population-specific Gaussian-process-like model. The first model shall be used to identify a vector which characterises the tasks. The second model, the main model, is created to consider the vector, which characterises the specific task, and to make predictions about the quantities of interest. The model should be able to perform this inference using a number of observed input-output variables $(\boldsymbol{x}^{e,i}, \boldsymbol{y}^{i})$. Such a consideration corresponds to the second part of the right-hand side of equation (13); i.e. the mutual information between the available training data and each task. Although in [32], the quantity is set to be minimised, in the current work it is believed that its maximisation would benefit the training of the population-based model. Maximisation of the aforementioned mutual information should yield a model which is able to identify the task from a few available data samples and use this information to effectively perform inference for unseen data.

Such a framework has been developed, termed *conditional neural processes* (CNP) in [42]. The framework aims at imitating the behaviour of a GP,

which uses some training data $\mathcal{D}^{tr} = \{(\boldsymbol{x}_1, \boldsymbol{y}_1), (\boldsymbol{x}_2, \boldsymbol{y}_2)..., (\boldsymbol{x}_N, \boldsymbol{y}_N)\}$, called *context points* in the original work, as its base to make predictions about new input variables $\boldsymbol{x}$ of interest. Although the basic formulation of a GP is considered to be a black-box model, it could be argued that it is not. As mentioned, a GP requires some training data as well as the selection of a *mean* and a *kernel* function [41]. The latter two functions can be defined by the analyser and can be parametric functions whose parameters can be learnt from the available data. However, the form of the kernel function has to be *a priori* defined, e.g. a Matern kernel, a squared-exponential kernel, a linear kernel or a periodic one. In order to select one of these functions, one would have to impose one's intuition and prior physical knowledge into the problem. However, if no prior knowledge exists, a general purpose kernel, such as a Gaussian kernel can be used, which is appropriate for regression and whose smoothness can be inferred or predefined.

The CNP neural network can be split in two subnetworks. The main network is the one that makes predictions given the varying environmental variables $\boldsymbol{x}^e$ and an embedding of the task-descriptive variables $\boldsymbol{x}^\tau$. The embedding of the task-descriptive variable is given by the second subnetwork, which infers an arbitrary set of task-descriptive variables exploiting information from pairs of inputs and outputs of a newly-presented task. More specifically, considering available data $\mathcal{D}^i = \{(\boldsymbol{x}_1, \boldsymbol{y}_1), (\boldsymbol{x}_2, \boldsymbol{y}_2)..., (\boldsymbol{x}_N, \boldsymbol{y}_N)\}$ for a task τ_i, then the CNP provides predictions of the quantity of interest given by,

$$\boldsymbol{y} = f^d(\boldsymbol{x}^e, \boldsymbol{x}_1, \boldsymbol{x}_2..., \boldsymbol{x}_N, \boldsymbol{y}_1, \boldsymbol{y}_2..., \boldsymbol{y}_N) = f(\boldsymbol{x}^e, \hat{\boldsymbol{x}}^\tau) \tag{19}$$

where $\boldsymbol{x}^e$ is the value of the environmental variables for which a prediction is needed, f^d is the CNP main network and $\hat{\boldsymbol{x}}^\tau$ is the output of the task-identification subnetwork. An example of such an assembly is shown in Figure 7. In the example, the subnetwork shown in orange lines is the task-identification subnetwork. In the specific case, the two available pairs of known input and output vectors for the newly-presented task are, $(\boldsymbol{x}^{e,1}, \boldsymbol{y}^{e,1})$ and $(\boldsymbol{x}^{e,2}, \boldsymbol{y}^{e,2})$. Although the model in the original work is not considered a meta-learning method, one could argue that the model essentially learns to perform the training procedure of a GP and can therefore be included in the meta-learning discipline. The model performs qualitatively the functionality described by equation (13), with a negative β and deals with the problem of parametrisation of the structures, in a more direct way than MAML.

Because the models are connected in such a manner, training such a model is done simply by backpropagating the error between the output and the target vectors. For the framework of the current paper, a CNP model can be trained by randomly sampling inputs to the task-identification subnetwork, input variables $\boldsymbol{x}^e$ and target values $\boldsymbol{y}$ from the data-rich structures. Then the models can be tested on structures for which only a few points are available

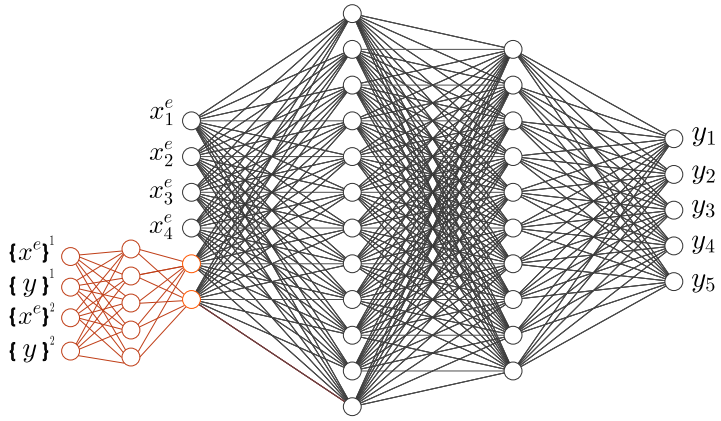


Figure 7: Neural network modelling the population with inputs regarding the external variables $\boldsymbol{x}^e$ and using a task identification subnetwork to map available data into some replacement variables for the task-descriptive variables.

and make predictions for new input variables. In contrast to MAML, the training of a CNP is done in a traditional neural-network-training manner. The dataset is formed by randomly sampling points from the available dataset and backpropagation is repeated for several epochs.

4. Applications

In order to evaluate the efficiency of the two discussed algorithms in creating models that learn based on the population physics, an application on simulated data is presented. The application refers to a population of simulated lumped-mass systems similar to the ones shown in Figure 1. Each structure of the population has a stiffness parameter k uniformly sampled from the interval $[8000, 12000]$. The structures of the population are excited at their first degree of freedom by a white noise signal and the simulations were performed using fourth-order Runge-Kutta integration. The temperature is taken into account as an environmental parameter affecting the structures. The way that temperature is affecting the structures is by increasing or decreasing the value of the stiffness parameter of the first three springs of the system. This is considered to be an imitation of the fact that structures are often heated disproportionally throughout their volume, for example a bridge is often heated more on its deck than its pillars because of the sun radiating on the top of the bridge.

Examples of *frequency response functions* (FRFs) of the first degree of freedom of a system of the population are shown in Figure 8. The variation of the FRFs is shown as different colours, from lower temperatures (blue curves) to higher temperatures (red curves). The relationship of the stiffness parameter k and the temperature is considered to be nonlinear and given by

$k = -13T^2 + 500T + 7200$ (Figure 9) and the temperature is considered to vary in the interval $[20, 40]$.

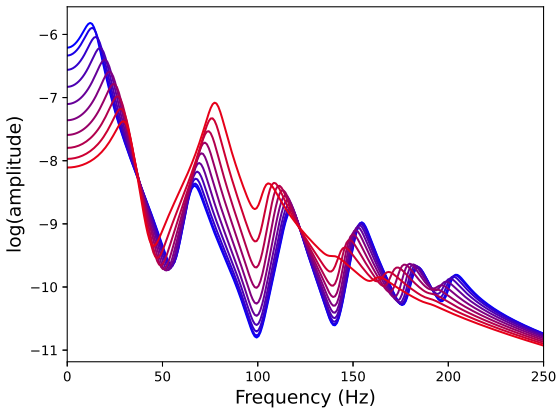


Figure 8: Frequency response functions of the first degree of freedom of a structure of the population for varying temperatures. Lower temperatures correspond to blue curves and as the temperature rises, the transition of the FRF is shown as a gradual change to red colour. Note that the changes are quite substantial because the purpose of the application is to illustrate the potential of the algorithms.

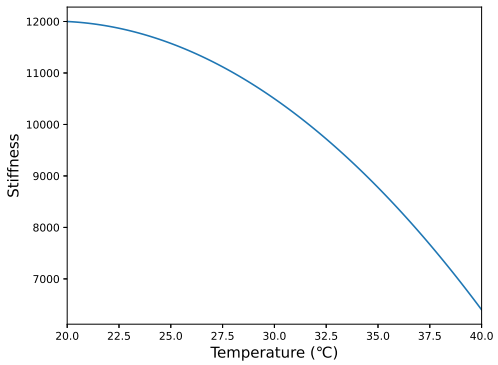


Figure 9: Relationship between the temperature and the stiffness of the first three springs of the systems of the population.

Three problems were studied in the current application. The first two refer to building neural network models that approximate the relationship between the temperature of the environment and the magnitude of a single spectral line from the FRFs. Two spectral lines were chosen for illustration, corresponding to 1 Hz and 50 Hz. Examples of the spectral lines for different temperatures and for three different structures are shown in Figure 10. The

two problems are studied because of a characteristic which is clear from the two plots. In the first case, the 1 Hz spectral lines, the relationship between the temperature and the task, and the value of the magnitude of the spectral line is a bijection. As a result, only one point suffices in order to characterise and identify the task for which inference is made; a functionality that both approaches should perform. In contrast, for the second case, overlapping between the curves is observed. Consequently it is expected that the latter shall be a more difficult task, because more than one point from each task is needed to perform the identification of the task.

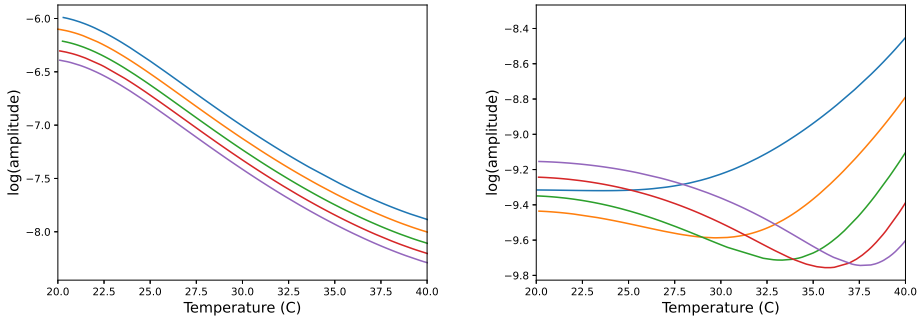


Figure 10: FRF line corresponding to frequency equal to 1 Hz (left) and 50 Hz (right), for values of the stiffness equal to 8000 (blue curve), 9000 (orange curve), 10000 (green curve), 11000 (red curve) and 12000 (curve).

The third problem is that of building a model to approach the whole FRF. To reduce the dimensionality of the FRFs, which are inferred, *principal component analysis* (PCA) [43] was performed on the samples. For completeness, samples of the population for varying temperature are shown in Figure 11. In the aforementioned figure, one can see the effect of temperature on the modelled quantity. Moreover, the physics of the population data-wise are revealed. The transformation of the curves for different structures is shown. It is worth noting that the described algorithms are expected to imitate a human capability; observing the curves of Figure 11, one could infer the values of the corresponding curve of a structure, for which only a few points (even just one point in this case) are available. This human functionality is about understanding the physics of the data or, in the current case, the physics of the population.

The population-based set-up is defined, considering a small subset of structures as the *training population*, for which data are freely available. Such a situation can be defined when one has extensively modelled a structure in operation or when one is able to test a structure in a laboratory for various excitations and environmental conditions. In contrast to other works on

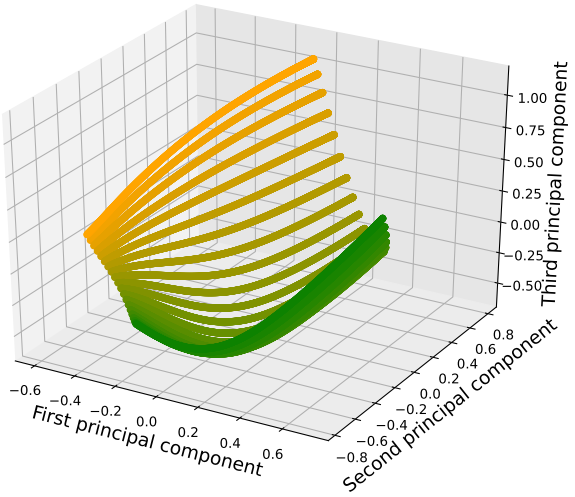


Figure 11: Principal components of samples of FRFs of members of the population. Points belonging to the same structure for different temperature are shown with the same colour. The colour difference refers to different structures of the population.

regression with meta-learning [36, 42, 44], where tasks are freely sampled during training from a distribution and the models are trained essentially based on an infinite number of available tasks, in the current work the number of training tasks (structures) is considered to be small but data for these structures are available for a large number of different input variables. Using these data, neural network models are trained following the described approaches. Then, the models are used to approximate the relationship between the temperature and the FRF of new structures - the *testing population* - for which only a few samples are available. The methods are then compared to a mainstream machine-learning approach, that of applying a *Gaussian process* (GP) [41] directly on the available data separately for every structure of the testing population. For the GP, a zero mean value function and a Gaussian kernel were selected. Different types of mean and kernel functions were not tested, because they are considered reasonable choices for a problem with a small amount of available data for which one has no prior intuition. The hyperparameters of the kernel function were optimised by maximising the likelihood of the model for the available data.

4.1. Application of MAML

The exact procedure followed to train neural networks with the MAML method is similar to a train-validate-test approach of standard machine

learning. In each case, a three-layer neural network is used, with one input layer, a hidden layer and an output layer. The activation function of the layers was a *hyperbolic tangent* (tanh) function for all layers. For $N+1$ structures in the training population, Algorithm 1 is used to train a neural network based on the data of N structures. The data from the structure which is left out and considered the *validation structure*, are used to optimise the hyperparameters of the model; in the current case the size of the hidden layer of the neural network. The sizes of the hidden layer considered were from the set $\{10, 20..., 100\}$. For each hidden-layer size, five random initialisations of the parameters of the network were considered and algorithm 1 was used to train the network. After every epoch of the algorithm, the model was adapted on the data of the validation structure; i.e. the model was trained by applying backpropagation using the data of the validation structure. The adapted model was tested on unseen-by-the-algorithm data from the validation structure. The instance of the model that achieved the lowest loss on the unseen data from the validation structure was considered the best model and was then tested on the structures from the testing population, whose number in all experiments was set equal to 200 structures. The whole procedure was repeated 50 times because of the random selection of the training population, which is expected to affect the performance of the model on the unseen structures of the population.

The training was repeated for different number of structures in the training population, as well as for different number of available samples for each testing structure. The model, which was previously trained using MAML, was adapted on samples coming from new testing structures and then the *normalised mean-squared error* (NMSE) was calculated for 200 values of input temperatures randomly sampled from the interval $[20, 40]$. The NMSE of the performance of the algorithm is defined by,

$$NMSE = \frac{100}{N\sigma_y^2}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2 \tag{20}$$

where $\hat{y}_i$ is the prediction of the model for the i^{th} input sample, y_i is the corresponding observation, σ_y is the standard deviation of the values of the observations y and N is the total number of samples. This NMSE is useful as a metric since it is equal to 100% if the model predictions ($\hat{y}_i$) are set to the mean value, i.e. $\hat{y}_i = \overline{y}$; values lower than 100% reveal that the model is indeed capturing correlations in the data. Experience with this NMSE indicates that good models are obtained for values of less than 5%, with a value of less than 1% for excellent models. The results presented here refer to the average NMSE of the 50 repetitions of training and testing, as well as to the standard deviation of the 50 population NMSEs.

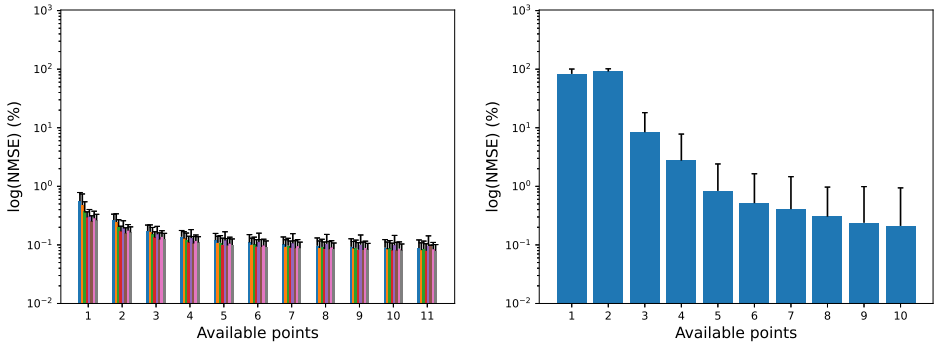


Figure 12: Average normalised mean-squared errors and corresponding error-bars for the first problem, for a testing population of 200 structures and for 100 data samples for each structure using a neural network trained via the MAML algorithm (left) and for a GP (right). On the left, the different colours represent MAML-trained neural networks with a training population of two (blue), three (orange), four (green), five (red), six (purple), seven (brown), eight (pink) and nine (grey) structures.

The results for the three problems are presented in Figures 12, 13 and 14; note that for the first and the third problems, the NMSE axis is shown on a logarithmic scale. As a first observation, one can easily see that the algorithm outperforms the GP approach in almost every case; this is to be expected, since the algorithm is trained based on data from the population, while the GP is not informed in any way by the population or by some prior physical knowledge of the analyst. However, for higher numbers of available training points, the GP is expected to achieve equal or lower NMSEs than the MAML algorithm, especially given that the presented problems are single-input single-output problems.

It can further be seen that the MAML algorithm functions as a traditional machine-learning algorithm regarding the number of available training data. The more available training structures, the lower the average error on the testing data and the lower the standard deviation of the errors in the population, indicating a more robust model. Small inconsistencies to these tendencies might be because of the random selection of the training structures, as well as the stochastic optimisation procedure of the neural-network models. This observation further encourages the belief that using the discussed algorithms may be a way to automatically extract exploitable knowledge from the population and use it to boost the performance of models for newly-presented structures. It is to the authors' knowledge that a GP can be informed by data from a population to boost its performance; however, such approaches may be included in the *grey-box* modelling discipline [45, 40], according to which, one imposes knowledge into the model; such knowledge

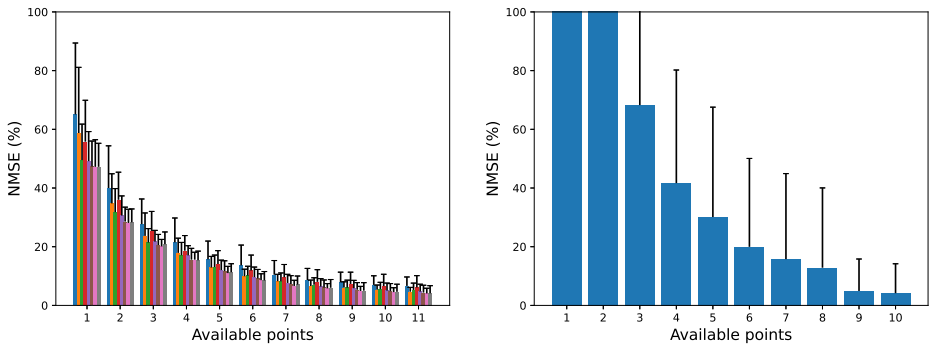


Figure 13: Average normalised mean-squared errors and corresponding error-bars for the second problem, for a testing population of 200 structures and for 100 data samples for each structure using a neural network trained via the MAML algorithm (left) and for a GP (right). On the left, the different colours represent MAML-trained neural networks with a training population of two (blue), three (orange), four (green), five (red), six (purple), seven (brown), eight (pink) and nine (grey) structures.

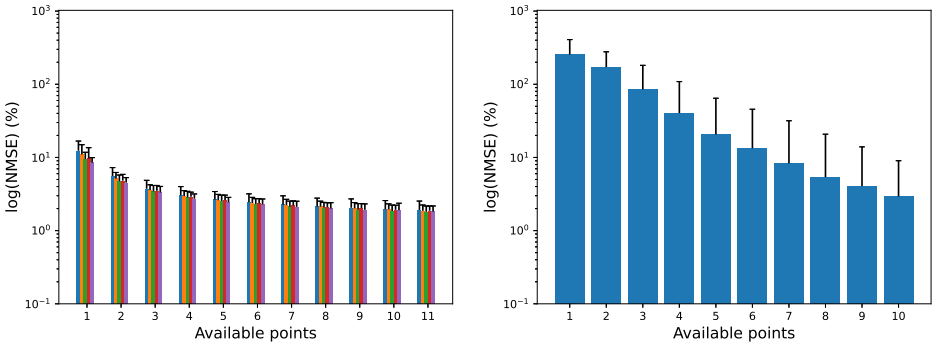
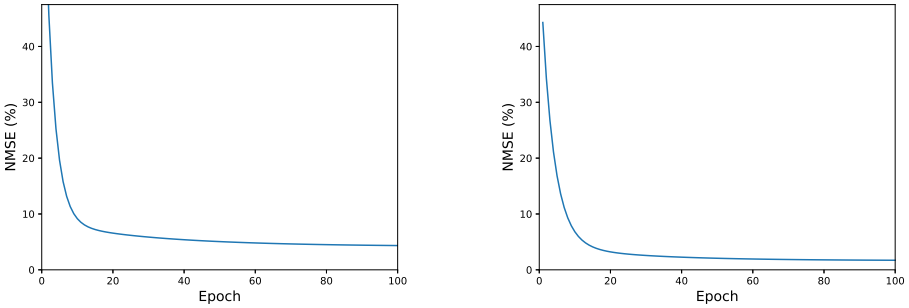


Figure 14: Average normalised mean-squared errors and corresponding error-bars for the third problem, for a testing population of 200 structures and for 100 data samples for each structure using a neural network trained via the MAML algorithm (left) and for a GP (right). On the left, the different colours represent MAML-trained neural networks with a training population of two (blue), three (orange), four (green), five (red), six (purple), seven (brown), eight (pink) and nine (grey) structures.



(a) (b)

Figure 15: Average normalised mean-square error history examples from the second problem, for (a) five training structures and five available samples for the testing structure and (b) nine training structures and seven available samples for the testing structure.

might also be acquired from a population of structures.

It is also worth examining the results even further; specifically the convergence of the neural-network model, which was trained using MAML. As mentioned in the original work, the model which is trained using MAML, is quite sensitive to different tasks. In the current case, different tasks refer to different structures. The average NMSE history for different sizes of the training population and available samples for the structures of the testing population are shown in Figure 15. The average NMSE refers to the 200 testing points of values of temperature uniformly sampled from the interval $[20, 40]$. It is clear that as the number of structures in the training population and the number of available samples for the testing structures increase, so does the convergence rate towards a minimum. This proportionality indicates that larger sizes of training populations allow the algorithm to learn better the physics of the population. Another important aspect of the training histories, is how stable they are. After some epochs, a minimum has been achieved and, even though the model is trained for longer than that, the error does not diverge significantly from the achieved minimum. The observed stability in the NMSE training histories also encourages the belief of the previous section, that using MAML, the trainable parameters of the model are "trapped" on a manifold, where the solutions for the population exist. In contrast to what is observed in the current figures, a model, whose trainable parameters would be freely allowed to adapt, would potentially start to overfit to the values of the training data and would not exhibit such a stable training history regarding the error on the testing data.

4.2. Application of CNP

The same framework was followed to test the performance of a CNP model on the population. Results are presented in a similar manner in Figures

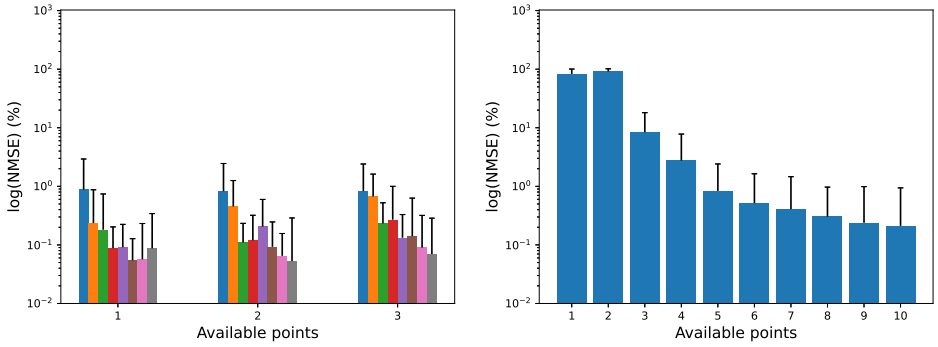


Figure 16: Average normalised mean-squared errors and corresponding error-bars for the first problem, for a testing population of 200 structures and for 100 data samples for each structure using a neural network trained via the CNP algorithm (left) and for a GP (right). On the left, the different colours represent CNP-trained neural networks with a training population of two (blue), three (orange), four (green), five (red), six (purple), seven (brown), eight (pink) and nine (grey) structures.

16, 17 and 18. As in the case of applying MAML, the algorithm also exhibits behaviour similar to traditional machine learning. As the number of available training structures increases, the error almost monotonically decreases. Similar to before, some inconsistencies are observed, which might be caused by the random selection of the training populations. The same behaviour is observed for the standard deviation of the error, which also decreases as more training structures are available, rendering a more robust model for the population.

The CNP model also appears to perform much better than the GP in the case of problems one and three, which, as discussed, is expected. It is also observed that for the second problem, the CNP approach performs better than MAML when more training structures are available. The better performance in such cases might be because of the different ways that the two models are trained. The MAML approach is an attempt to find an initialisation point for the trainable parameters of the neural network and from there to quickly fit a model to the task-specific optimal parameters. MAML is therefore an implicit way of identifying the task via the error signals passed by the backpropagation process to the model. On the contrary, CNP is optimised exclusively with a view to identifying the task from the available data, inferring a task-descriptive vector and using this vector as a feature to make predictions for new inputs. It appears that MAML, where backpropagation is performed through the optimisation algorithm, is a more complicated attempt to learn the physics of the population and, as a result, the more direct approach of CNP in this case outperforms MAML.

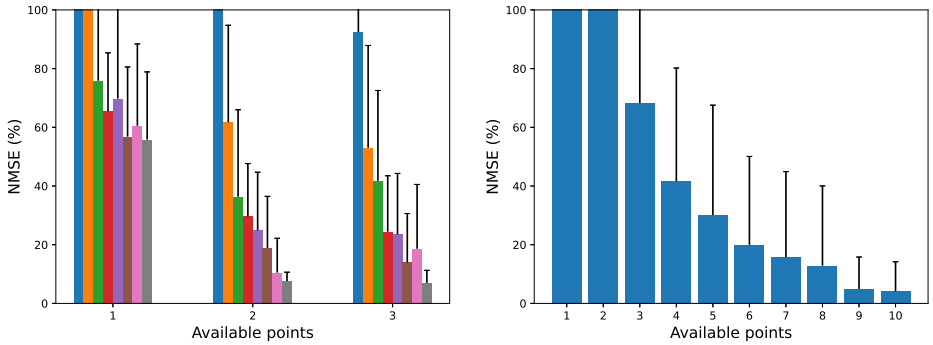


Figure 17: Average normalised mean-squared errors and corresponding error-bars for the second problem, for a testing population of 200 structures and for 100 data samples for each structure using a neural network trained via the CNP algorithm (left) and for a GP (right). On the left, the different colours represent CNP-trained neural networks with a training population of two (blue), three (orange), four (green), five (red), six (purple), seven (brown), eight (pink) and nine (grey) structures.

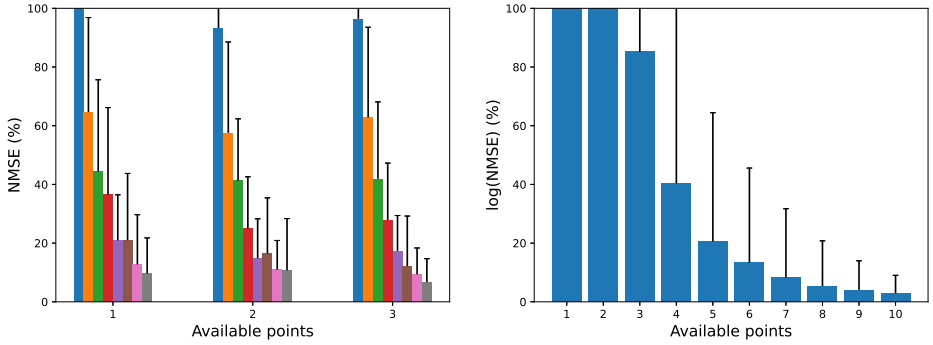


Figure 18: Average normalised mean-squared errors and corresponding error-bars for the third problem, for a testing population of 200 structures and for 100 data samples for each structure using a neural network trained via the CNP algorithm (left) and for a GP (right). On the left, the different colours represent CNP-trained neural networks with a training population of two (blue), three (orange), four (green), five (red), six (purple), seven (brown), eight (pink) and nine (grey) structures.

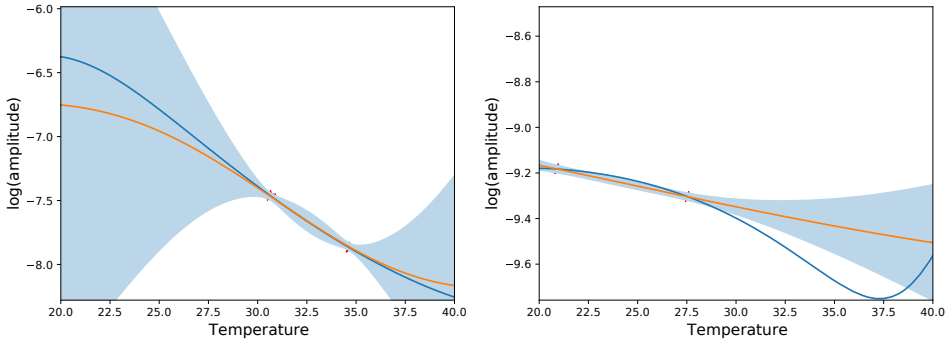


Figure 19: The mean of the predictions (orange curve) and confidence intervals of ± 3 standard deviations of the predictions (shaded blue area) of a GP fitted to available data from a testing structure (red star points) for the 1 Hz problem (left) and the 50 Hz problem (right). The real underlying relationship is shown as the blue curve and the predictions of the model as the orange curve.

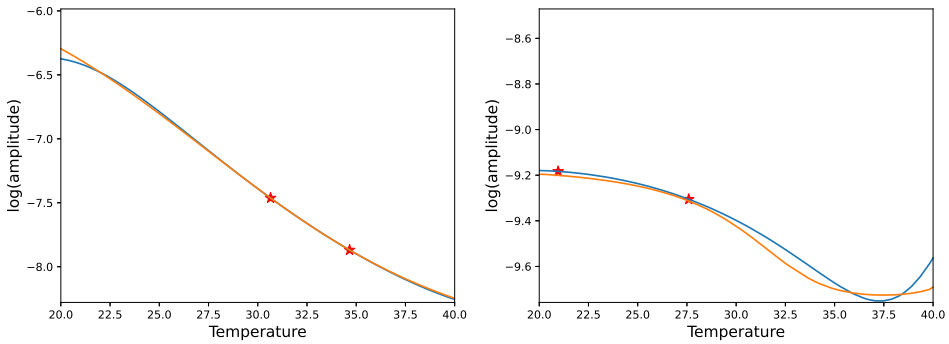


Figure 20: Example of fitting a neural network, trained according to the MAML algorithm on nine training structures, to available data from a testing structure (red star points) for the 1 Hz problem (left) and the 50 Hz problem (right). The real underlying relationship is shown as the blue curve and the predictions of the model as the orange curve.

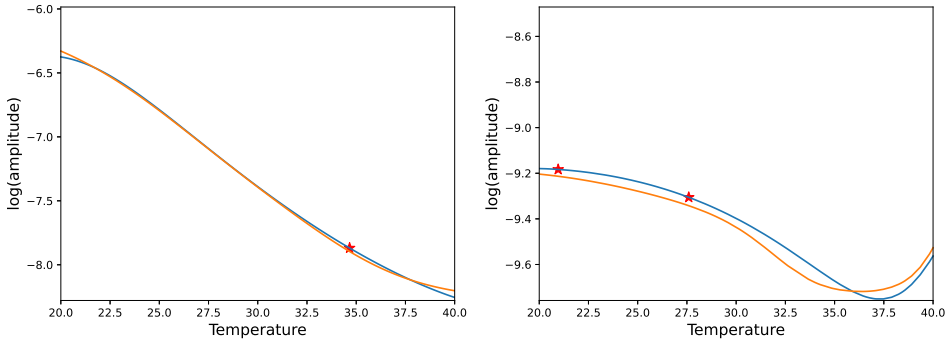


Figure 21: Example of the performance of a CNP model, trained using data from nine training structures. The algorithm uses data from testing structure (red star points) to make predictions for the 1 Hz problem (left) and the 50 Hz problem (right). The real underlying relationship is shown as the blue curve and the predictions of the model as the orange curve.

To better understand why population-informed models outperform the traditional machine-learning approach, some examples of fitting these models to a small subset of data from a testing structure are shown in Figures 19, 20 and 21. For the current examples, the population-informed models are trained using a training population of nine structures. In the figures, examples of the performance of the models for the first two problems are presented. Observing the behaviour of the population-informed models away from the available samples gives a clear indication that these models have incorporated part of the physics of the structures. For CNP it is also clear that with only one available sample from the testing structure, the model is able to almost perfectly approximate the underlying relationship, although, as discussed the first problem is quite a simple problem. The GP on the other hand is not informed in any way from the population. As a result, its predictions are not as efficient as the predictions of the other two models.

5. Conclusions and next steps

The current work aims at motivating the creation of data-driven models, which are forced to respect the underlying physics of some population of problems. The physics of the population are to be learnt via the use of data from a population of structures. The desired result is inspired by the functionality of physics-based models, which utilise parameters that describe the characteristics of various structures. These parameters belong to a predefined interval, in contrast to the trainable parameters of a data-driven model, whose domain is the whole set of real numbers.

Two approaches are studied for the purpose of defining population-informed

models. The first is based on forcing the trainable parameters of a neural network model to lie on a manifold, where minima for modelling structures of the population exist. The approach is implemented using a meta-learning technique for training neural networks, the model-agnostic meta-learning algorithm (MAML). The second approach is based on using a set of available input-output observations from a structure to identify the structure for which inference is to be made. The approach is implemented via the conditional neural processes algorithm (CNP), which exploits a task-identification sub-network to infer a task-descriptive vector, from a small subset of available task-specific data, and use this vector as a feature in the main neural network model to make predictions about input-variables of interest.

The two types of population-informed models are tested based on three simulated datasets. The datasets comprise FRFs of simulated lumped-mass structures as a function of the temperature of the environment of the structures. The two first datasets refer to the prediction of the magnitude of single spectral lines of the FRF of the lumped-mass systems and the third problem is about inferring the whole FRF. The difference between the first two problems is that for the first, a one-to-one relationship exists between the structure (task) and the temperature, and the magnitude of the spectral line, while for the second task overlapping between the task curves exists.

The algorithms in all cases are tested on a small population of available structures, in contrast to other works on meta-learning where one had unrestricted access to random tasks during training. Moreover, because the selection of the training structures naturally affects the result, the training and testing of the algorithms were performed several times and the average NMSE and the standard deviation of the NMSEs were calculated. The performance of the two algorithms was compared to a traditional machine-learning method - a Gaussian process (GP) - which is suitable for cases of modelling datasets with only a few available data.

The results reveal that both algorithms are able to exploit knowledge from the population. The algorithms are able to perform better than the GP, which, in the current work, is not informed by the population or prior knowledge of the analysers. Both MAML and CNP exhibit low average population errors for the second case study and very low average population errors for the first and third problems, where the relationship between the temperature and the structure, and the quantity of interest is a bijection. Moreover, both algorithms seem to behave similarly to a traditional machine-learning algorithm regarding the number of available training structures and the average error across the testing population as well as the standard deviation of these errors. It is clear that the more training structures one has, the more accurate and robust the model is within the population. It is also worth noting, that the current approach is a completely physics-blind approach, in

the sense that one does not need to induce any kind of physical knowledge in the algorithm for them to function properly. In future work, it would be interesting to examine a combined approach, including some physics into the procedure.

Although it is not the algorithms' primary objective, they could potentially be used to identify damaged structures. Including data from structures with different levels of damage in the population, the algorithm could learn to make inferences for such situations. Thus, with a few data from a structure, one could infer its behaviour for a wider range of environmental or operational conditions. Making such inferences, the comparison between the healthy states and the testing states could be made more efficiently.

The algorithms are tested only on simulated data. However, the current work aims at motivating the use of black-box models in a similar manner to physics-based models. A major ability of physics-based models, is their reusability. The models can be built for a specific structure and then slightly modified to be used on another. To bridge the gap between the use of the two types of models, the approach herein is to define neural network models which are built in order to perform inference within a population. The two presented methods appear to create models suitable for such applications. Further validation of the methods on experimental data is needed; however, the results are encouraging. It is also believed that such models may add extra credibility to the use of data-driven models, because their behaviour and their goal is better defined in terms of being models of a population, rather than models that are only able to perform on a single task. Models defined in the described ways could more easily be trusted, as they can be validated for a set of structures of the population. Because they draw information from a population, the models do not begin their learning from complete ignorance of the underlying physics of the population, which renders them as not completely physics-ignorant models and makes their behaviour more trustworthy than traditional machine-learning methods.

Acknowledgements

The authors wish to gratefully acknowledge support for this work through grants from the Engineering and Physical Sciences Research Council (EPSRC), UK, via the Programme Grant EP/R006768/. For the purpose of open access, the authors have applied a Creative Commons Attribution (CC BY) licence to any Author Accepted Manuscript version arising.

References

[1] K.-J. Bathe. *Finite Element Procedures.* Klaus-Jurgen Bathe, 2006.

[2] P. Huthwaite. Accelerated finite element elastodynamic simulations using the GPU. *Journal of Computational Physics*, 257:687–707, 2014.

[3] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, Inc., New York, NY, USA, 1995.

[4] C.M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag, 2006.

[5] K. P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012.

[6] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*, volume 1. MIT press Cambridge, 2016.

[7] E.J. Cross, S.J. Gibson, M.R. Jones, D.J. Pitchforth, S. Zhang, and T.J. Rogers. Physics-informed machine learning for structural health monitoring. In *Structural Health Monitoring Based on Data Science Techniques*, pages 347–367. Springer, 2022.

[8] U.T. Tygesen, K. Worden, T.J. Rogers, G. Manson, and E.J. Cross. State-of-the-art and future directions for predictive modelling of offshore structure dynamics using machine learning. In *Dynamics of Civil Structures, Volume 2*, pages 223–233. Springer, 2019.

[9] C.R. Farrar and K. Worden. *Structural Health Monitoring: A Machine Learning Perspective*. John Wiley and Sons, 2011.

[10] P. Gardner, X. Liu, and K. Worden. On the application of domain adaptation in structural health monitoring. *Mechanical Systems and Signal Processing*, 138:106550, 2020.

[11] M. Raissi, P. Perdikaris, and G.E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

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

[13] J. Gosliga, P.A. Gardner, L.A. Bull, N. Dervilis, and K. Worden. Foundations of population-based SHM, part II: Heterogeneous populations–Graphs, networks, and communities. *Mechanical Systems and Signal Processing*, 148:107144, 2021.

[14] P. Gardner, L.A. Bull, J. Gosliga, N. Dervilis, and K. Worden. Foundations of population-based SHM, part III: Heterogeneous populations–mapping and transfer. *Mechanical Systems and Signal Processing*, 149:107142, 2021.

[15] G. Tsialiamanis, C. Mylonas, E. Chatzi, N. Dervilis, D.J. Wagg, and K. Worden. Foundations of population-based SHM, Part IV: The geometry of spaces of structures and their feature spaces. *Mechanical Systems and Signal Processing*, 157:107692, 2021.

[16] L.A. Bull, M. Dhada, O. Steinert, T. Lindgren, A.K. Parlikad, A.B. Duncan, and M. Girolami. Knowledge transfer in engineering fleets: Hierarchical Bayesian modelling for multi-task learning. *arXiv preprint arXiv:2204.12404*, 2022.

[17] J. Vanschoren. Meta-learning: A survey. *arXiv preprint arXiv:1810.03548*, 2018.

[18] T. Hospedales, A. Antoniou, P. Micaelli, and A. Storkey. Meta-learning in neural networks: A survey. *arXiv preprint arXiv:2004.05439*, 2020.

[19] P. Gardner, L.A. Bull, N. Dervilis, and K. Worden. On the application of kernelised Bayesian transfer learning to population-based structural health monitoring. *Mechanical Systems and Signal Processing*, 167:108519, 2022.

[20] P. Gardner, L.A. Bull, J. Gosliga, J. Poole, N. Dervilis, and K. Worden. A population-based SHM methodology for heterogeneous structures: Transferring damage localisation knowledge between different aircraft wings. *Mechanical Systems and Signal Processing*, 172:108918, 2022.

[21] D.S. Brennan, T. Rogers, E.J. Cross, and K. Worden. On quantifying the similarity of structures via a graph neural network for population-based structural health monitoring. In *Proceedings of ISMA2022 including USD2022*. KU Leuven, 2022.

[22] K. Worden, E.J. Cross, R.J. Barthorpe, D.J. Wagg, and P. Gardner. On digital twins, mirrors, and virtualizations: Frameworks for model verification and validation. *ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems Part B Mechanical Engineering*, 6(3), 2020.

[23] B.F. Schutz. *Geometrical Methods of Mathematical Physics*. Cambridge University Press, 1980.

[24] M.J.D. Hamilton. *Mathematical Gauge Theory*. Springer, 2017.

[25] Y. Wang and Q. Yao. Few-shot learning: A survey. 2019.

[26] G. Koch, R. Zemel, R. Salakhutdinov, et al. Siamese neural networks for one-shot image recognition. In *International Conference on Machine Learning Deep Learning workshop*, volume 2. Lille, 2015.

[27] E. Hoffer and N. Ailon. Deep metric learning using triplet network. In *International Workshop on Similarity-based Pattern Recognition*, pages 84–92. Springer, 2015.

[28] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra, et al. Matching networks for one shot learning. *Advances in Neural Information Processing Systems*, 29, 2016.

[29] M. Oquab, L. Bottou, I. Laptev, and J. Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[30] S.-X. Chen, L. Zhou, and Y.-Q. Ni. Wheel condition assessment of high-speed trains under various operational conditions using semi-supervised adversarial domain adaptation. *Mechanical Systems and Signal Processing*, 170:108853, 2022.

[31] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[32] M. K. Titsias, F. J.R. Ruiz, S. Nikoloutsopoulos, and A. Galashov. Information theoretic meta learning with gaussian processes. In *Uncertainty in Artificial Intelligence*, pages 1597–1606. PMLR, 2021.

[33] S. Ruder. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*, 2017.

[34] R. Caruana. Multitask learning: A knowledge-based source of inductive bias. In *Proceedings of the Tenth International Conference on Machine Learning*, pages 41–48. Citeseer, 1993.

[35] L. Duong, T. Cohn, S. Bird, and P. Cook. Low resource dependency parsing: Cross-lingual parameter sharing in a neural network parser. In *Proceedings of the 53rd annual meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: short papers)*, pages 845–850, 2015.

[36] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, pages 1126–1135. PMLR, 2017.

[37] B. Lake, R. Salakhutdinov, J. Gross, and J. Tenenbaum. One shot learning of simple visual concepts. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 33, 2011.

[38] M. Cheng, C. Dang, D.M. Frangopol, M. Beer, and X.-X. Yuan. Transfer prior knowledge from surrogate modelling: A meta-learning approach. *Computers and Structures*, 260:106719, 2022.

[39] J. Berner, P. Grohs, G. Kutyniok, and P. Petersen. The modern mathematics of deep learning. *arXiv preprint arXiv:2105.04026*, 2021.

[40] D.J. Pitchforth, T.J. Rogers, U.T. Tygesen, and E.J. Cross. Grey-box models for wave loading prediction. *Mechanical Systems and Signal Processing*, 159:107741, 2021.

[41] C.E. Rasmussen. Gaussian processes in machine learning. In *Summer School on Machine Learning*, pages 63–71. Springer, 2003.

[42] M. Garnelo, D. Rosenbaum, C. Maddison, T. Ramalho, D. Saxton, M. Shanahan, Y.W. Teh, D. Rezende, and S.M.A. Eslami. Conditional neural processes. In *International Conference on Machine Learning*, pages 1704–1713. PMLR, 2018.

[43] S. Wold, K. Esbensen, and P. Geladi. Principal component analysis. *Chemometrics and Intelligent Laboratory Systems*, 2(1-3):37–52, 1987.

[44] M. Andrychowicz, M. Denil, S. Gomez, M. W. Hoffman, D. Pfau, T. Schaul, B. Shillingford, and N. De Freitas. Learning to learn by gradient descent by gradient descent. *Advances in Neural Information Processing Systems*, 29, 2016.

[45] E.J. Cross, T.J. Rogers, and T. J. Gibbons. Grey-box modelling for structural health monitoring: physical constraints on machine learning algorithms. *International Workshop on Structural Health Monitoring*, 2019.