Systematic sensor placement for structural anomaly detection in the absence of damaged states

Caterina Bigoni[a,*], Zhenying Zhang[a], Jan S. Hesthaven[a]

[a] *Ecole Polytechnique Fédérale de Lausanne (EPFL), Switzerland*

May 7, 2020

Abstract

In structural health monitoring (SHM), risk assessment and decision strategies rely primarily on sensor responses. Simulated data can be generated to emulate the monitoring phenomena under different natural operational and environmental conditions in order to discriminate relevant features and thus identify potential anomalies. Reduced order modelling techniques and one-class machine learning algorithms allow to efficiently achieve this goal for a fixed number and location of sensors. However, since the number of sensors available on a structure is often a limitation for SHM, identifying the optimal locations that maximize the observability of the discriminant features becomes a fundamental task. In this work we propose to use the variational approximation of sparse Gaussian processes to systematically place a fixed number of sensors over a structure of interest. The healthy parametric variations of the structure are included by clustering the inducing inputs, i.e., the outcome of variational inference. This technique is tested on several numerical examples and is demonstrated to be efficient in detecting damages. In particular, it allows for considering the realistic case where damage types and locations are *a priori* unknown, thus, overcoming the main limitation of existing sensor placement strategies for SHM.

Keywords: Sensor placement, anomaly detection, sparse Gaussian processes, variational inference, structural health monitoring (SHM)

1. Introduction

When monitoring a structure over time, its deterioration and damages represent a great concern and the early detection of unexpected behaviour might prevent sudden shutdowns or help avoid catastrophes. In the last decades, the traditional visual inspection of complex and valuable assets such as bridges, buildings, wind turbines, etc., has been gradually replaced with structural health monitoring (SHM) systems, which aim at providing reliable information on the performance and integrity of a structure [14]. In the context of SHM, the combination of sensor measurements, numerical models simulating the underlying behaviour of a structure of interest under different environmental and operational conditions, and machine learning techniques has led to the design of structural *digital twins*. These virtual representations seek to assess the structural state of damage in real-time and can potentially support an automated decision-making strategy. Even though there exists a variety of SHM techniques, mainly differing by the quantity of interest to estimate or for the type of sensors employed while keeping into account the different requirements and limitations, they all rely on a network of sensors. Hence, their performance depends critically on the quality of the information collected at those sensors. Clearly, both improving sensor sensitivity and deciding where to place sensors play a key role in the digital twin industry.

Motivated by the opportunities of cost reduction for SHM systems and the improvement in the quality of the monitoring outcome, optimization of sensor placement (OSP) has received growing interest during the

*Corresponding author.
E-mail address: caterina.bigoni@epfl.ch

last decades. The exhaustive review [33] provides a collection of examples of OSP applied to SHM, classified based on the different techniques employed for the sensor placement optimization itself, among which the vibration-based and the wave-based monitoring are the most commonly used. While the former depend on the dynamics of the structure using passive sources, e.g., only the ambient loads on the structures are considered, the latter are usually used in the active sensing domain. Where as vibration monitoring techniques aim at identifying changes in the natural frequencies and mode shapes with respect to a baseline, in the wave-based monitoring field, vibrations are generated by a controlled source, e.g., a sinusoidal wave or a short pulse impulse, and signal-processing techniques are used to differentiate baseline time-dependent responses from the reflections and refraction of the wave caused by the presence of damages. Since the non-destructive impulses used to excite a structure have a high damping effect, i.e., it is difficult to observe the effect of the guided-wave far from the source, wave-based monitoring techniques are usually employed to monitor pipes or plate-like components with complex geometries, e.g., in aeronautical applications [31, 49]. On the contrary, large-scale assets, e.g., dams, bridges, etc., are usually monitored by vibration-based techniques, see e.g., [7], or by static approaches, see e.g., [22].

Despite their fundamental differences, the general deployment of an OSP strategy is similar for both approaches. The OSP process can be split into a sequence of a few stages going from the choice of sensor types, over to the definition of operational parameters, e.g., the candidate sensor locations, and, finally, to the characterization of a suitable cost function and optimization algorithm, e.g., gradient-based techniques are chosen when the cost function is continuous and differentiable, while meta-heuristic optimizations might be necessary otherwise. We discuss here the state of the art of OSP for both the vibration- and the wave-based monitoring techniques. Among the most popular placement strategies for the former, we note the effective independence method (EFI), the kinetic energy method (KE), and the more recent information theory approach, which obtains an optimal placement of sensors by minimizing the information gain within a Bayesian experimental design framework, see e.g., [34, 7, 3]. For active sensing based on guided waves, we focus on [15] and [28]. In the former, the authors propose an optimization procedure where the sensor locations are chosen to minimize the appearance of false alarms and mis-detections. The latter proposes a strategy to increase the sensitivity to damage by using simulation-based techniques, in which, by comparing the numerical solution of the guided-wave propagation in undamaged versus damaged scenarios, sensors are placed where the largest increase in the signal amplitude is observed. When the wave propagation patterns are very complicated, it has been proposed to maximize the area of coverage (MAC) within a sensor network, see e.g., [49], where physical properties of Lamb wave propagation and complex geometrical properties are taken into account, or [47], where the ellipse equations with the sensor actuator pair as the foci are used to compute the coverage area.

We note that, with the exception of the strategies which maximize the coverage area, all OSP techniques require knowledge about the characteristics of the damage, e.g., its type, its location, its severity, or its size. Consequently, these approaches do not generalize well when other types of damages occur and, even though engineering knowledge can certainly direct the attention to damages that are more likely to occur, it seems unreasonable to characterize them all. In particular, when relying on numerical simulations to describe the effect of a particular damage on a structure, including many damage types and all possible combinations becomes computationally intractable. A valid alternative is to resort to anomaly detection techniques, where damages are identified only by looking at the output of multiple undamaged scenarios, collected under different standard conditions, which may represent environmental or operational healthy variations. We refer to [29, 5] and references therein for a description on how to address the damage detection problem with anomaly detection learning strategies for a fixed network of sensors. However, many questions arise if one wishes to find the optimal sensor locations in the absence of any damage information. In particular, the definition of new operational parameters and their corresponding cost function must be considered.

In this work we propose a novel strategy for sensor placement in the context of anomaly detection applied to SHM when a fixed budget is given, i.e., the number and type of sensors is fixed. The sensor locations are systematically identified as the spacial positions for which the reconstruction error of an output of interest at all *unsensed* locations is minimized. The quantity of interest chosen to define the cost function for the sensor placement optimization algorithm is the same quantity used to train the anomaly detection classifier which distinguishes healthy configurations from damaged ones. As such, the proposed placement strategy is based on an appropriate indicator of the damage detection performance of a given network. More precisely, we employ the variational inference of sparse Gaussian process regression (GPR) for a damage-sensitive quantity

of interest representing an healthy scenario, and we use the inducing inputs as the sensor locations. With the variational formulation, sensor locations are selected by minimizing the Kullback-Leibler (KL) divergence between the exact posterior distribution and the variational distribution. Therefore, placing sensors at the corresponding location of the inducing inputs addresses both the information compression of the whole domain and the total variance reduction at the sensor locations. We also rely on an Expectation-Maximization (EM)-like algorithm for the training phase, which, on one hand, prevents a combinatorial search in the case of a discrete admissible set of points and, on the other hand, allows us to include domain restrictions in the optimization to avoid placing sensors in areas difficult to reach or not suitable for monitoring. Furthermore, we extend the proposed algorithm to take into account the natural variations of the model parameters, e.g., loads, boundary conditions, material properties, etc., by means of an unsupervised clustering algorithm. To conclude, we present some numerical examples to test the validity of the proposed method. In particular, we resort to a wave-propagation based strategy to place sensors on both 2D and 3D structures and to a static monitoring approach with passive sources to place sensors on a 3D representation of an offshore jacket.

We observe that we can relate some features of our approach to existing methods which are not specifically designed for SHM. First, the choice of recurring to GPs for sensor placement has been proposed in [10, 25], where either the maximum entropy principle or a mutual information criterion are used to identify near-optimal locations. In contrast, our work replaces the classic GPR model with a sparse variational approximation, which at the same time identifies the optimal sensors as the inducing points automatically and accommodates problems with large data set. Additionally, the strategy presented in [25] is used to monitor diffusion-like spatial phenomena, e.g., temperature in an indoor environment, while the SHM applications involve more complex phenomena, for which the training of a GPR is not always straightforward. Second, in the recent work [2], the authors propose a strategy to place sensors in a systematic manner to assist field experts in placing sensors in nuclear reactors. In particular, they propose to use the magic points found by the greedy algorithm of the generalized empirical interpolation method (GEIM) as sensor locations and show the effectiveness of this strategy on multidimensional examples based on synthetic measurements. Lastly, sparse approaches for sensor placement have been proposed in [6], where the authors exploit the low-dimensional structure exhibited by many high-dimensional systems to compress a signal to very few measurements if the sole objective is classification. Despite the use of sparsity-promoting techniques, this work is entirely based on classification, which is different from the scope of our work.

The remainder of this paper is organised as follows. Section 2 presents the physical phenomena and synthesizes how we efficiently construct a database of healthy configurations in both a dynamic and a static scenario. Sparse Gaussian process approximations are presented in Section 3 with a particular emphasis on variational sparse GPR. We explain how variational approximations are used for sensor placement in the absence of damage states in Section 4 and provide numerical evidence of the quality of this method in Section 5. Conclusions are given in Section 6.

2. Generating a database of synthetic healthy measurements

Simulation-based strategies provide a tool to monitor a structure of interest where experimental measurements are replaced with synthetic sensor signals, thus allowing to generate accurate datasets inclusive of many possible scenarios, which would be otherwise unrepresented. As both practical and efficient techniques, they have received increasing attention in recent years, see e.g., [27, 48, 37, 5, 24, 40]. Although a key step in SHM corresponds to the identification of good locations to place sensors, classic simulation-based strategies for damage detection often rely on the assumption that these locations are known, i.e., the structure of interest is already equipped with a network of sensors. As mentioned in Section 1 and further clarified in Section 4, the placement strategy proposed here is based on the same quantity of interest used to define damage detection classifiers. As a direct consequence, the practical process of generating a synthetic database, used either for anomaly detection or for sensor placement, is the same. Hence, in this work we focus on the construction of a database of simulated healthy configurations where a few given sensor locations are replaced with the points of a coarse mesh over the domain of interest. The optimal locations will be chosen as a subset of these points or as an arbitrary new set which belongs to the initial domain in a way that will be specified in Section 4.

In the remaining of this section, we first provide a short summary of anomaly detection strategies in Section 2.1. Then, in Section 2.2, we present the mathematical formulation of the governing physical problem, i.e.,

the parametric acoustic-elastic equation in both its dynamic form and its simplified static version, together with its numerical discretization. The explanation on how to efficiently deal with the need of repeatedly solving the problem for multiple parameters using the reduced basis method is also explained. We conclude with Section 2.3, where we define the chosen quantity of interest, obtained by extracting damage-sensitive features from the raw signals.

2.1. A brief recap of SHM anomaly detection

Different from a supervised learning approach, in the anomaly detection framework, the dataset does not include any damage scenarios. This is done under the assumption that since it would be unreasonable to describe all types of damages, representing only some damaged configurations would lead to a bias towards certain types and therefore to mis-detections with high probability. Classic supervised learning algorithms, where every different damage type is associated with a different categorical class, are here replaced with semi-supervised learning techniques, where only healthy states are used to train *one-class classifiers*, e.g., one-class support vector machines, local outlier factor, or auto-encoders. We note that, to avoid redundancies, in the context of both one-class and standard classification, raw measurements, e.g., displacements or accelerations, are not directly used in the training, but instead they are processed into features which are sensitive to damages but robust to noise and healthy variations. Then, in the online phase, the classifier is tested against new measurements to assess if they conform to the normal condition, reflected in the offline data, i.e., test samples will be classified either as healthy (inlier) or unhealthy (outlier).

We observe that, with anomaly detection techniques it is no longer possible to classify damages by type. However, by training a separate one-class classifier for each separate location, damage localization and severity can still be assessed for a given array of sensors. We refer the interested reader to [8] for a thorough description of outlier detection algorithms and to [29, 5] and references therein for how such algorithms are used in the context of SHM.

2.2. The governing problem of linear elasticity

Let $\Omega \subset \mathbb{R}^d$ with $d = \{2, 3\}$ be an open bounded domain, approximating the geometry of a given structure of interest and let $[0, T]$ be a relevant time domain for sensor measurements. Let us also consider a p-dimensional parameter space $\Omega_\mu = [\mu_1^1, \mu_2^1] \times [\mu_1^2, \mu_2^2] \times \cdots \times [\mu_1^p, \mu_2^p] \subset \mathbb{R}^p$, representing the baseline variations of healthy configurations under normal environmental and operational conditions, which can be described by both physical and geometrical properties. For a given parameter $\boldsymbol{\mu} = [\mu^1, \ldots, \mu^p] \in \Omega_\mu$, we seek the vector-valued displacement $\boldsymbol{u} = \boldsymbol{u}(\boldsymbol{x}, t; \boldsymbol{\mu}) : \Omega \times [0, T] \times \Omega_s \to \mathbb{R}^d$ such that

$$\rho \frac{\partial^2 \boldsymbol{u}}{\partial t^2} + \rho\eta \frac{\partial \boldsymbol{u}}{\partial t} - \nabla \cdot \boldsymbol{\sigma}(\boldsymbol{u}; \boldsymbol{\mu}) = s(\boldsymbol{x}, t; \boldsymbol{\mu}) \quad \text{in } \Omega \times (0, T]. \tag{1}$$

In the above strong-form formulation, ρ is the density, η is a non-dimensional damping coefficient, $\boldsymbol{\sigma} = \boldsymbol{\sigma}(\boldsymbol{u}; \boldsymbol{\mu})$ is the stress tensor $\boldsymbol{\sigma} = 2\mu\boldsymbol{\varepsilon}(\boldsymbol{u}) + \lambda \mathrm{Tr}\left(\boldsymbol{\varepsilon}(\boldsymbol{u})\right) \mathbb{I}$, where $\mathbb{I}$ is the d dimensional identity matrix, $\mathrm{Tr}(\cdot)$ is the trace operator applied to the strain tensor $\boldsymbol{\varepsilon}(\boldsymbol{u}) = \frac{1}{2}\left(\nabla \boldsymbol{u} + (\nabla \boldsymbol{u})^T\right)$ and the Lamé constants μ and λ are defined by E, the Young's modulus, and ν, the non-dimensional Poisson's ratio, as

$$\mu = \frac{E}{2(1+\nu)} \quad \text{and} \quad \lambda = \frac{E\nu}{(1+\nu)(1-2\nu)}. \tag{2}$$

Equation (1) is equipped with suitable boundary and initial conditions, which may depend on $\boldsymbol{\mu}$, and $s = s(\boldsymbol{x}, t; \boldsymbol{\mu})$ is a parameter-dependent function $s : \Omega \times (0, T] \times \mathcal{P} \to \mathbb{R}^d$ representing the source term.

After introducing a suitable spatial and temporal discretization, Equation (1) can be solved numerically, by resorting for example to the finite element (FE) method. The continuous solution $\boldsymbol{u}(t; \boldsymbol{\mu})$ of the weak-form of (1) is therefore replaced with its discrete counterpart $\boldsymbol{u}_h(t_n; \boldsymbol{\mu}) \in V_h$, where V_h is a conforming finite-dimensional subspace of $V = H^1(\Omega; \mathbb{R}^d)$ with $\dim(V_h) = N_h$. Moreover, $t_n = n\frac{T}{N_t}$ is the n-th time step of the discrete time interval $[0, T]$, which is partitioned into N_t equal sub-intervals. With the goal of sensor placement, we are only interested in the solution at few specific locations, representing the vertices of a coarse mesh with n_{dof} degrees of freedom. The parametric discrete displacement signal $\mathbf{u}_i(\boldsymbol{\mu})$ are $(N_t+1)\times d$-dimensional vectors defined as

$$\mathbf{u}_i(\boldsymbol{\mu}) := [\mathbf{u}_i^{\boldsymbol{\mu}}(t_0), \mathbf{u}_i^{\boldsymbol{\mu}}(t_1), \ldots, \mathbf{u}_i^{\boldsymbol{\mu}}(t_{N_t})] \quad \text{for } i = 1, \ldots, n_{\mathrm{dof}}, \tag{3}$$

where $\mathbf{u}_i^{\boldsymbol{\mu}}(t_n) = \boldsymbol{u}_h(\boldsymbol{x}_i, t_n; \boldsymbol{\mu}) = \sum_{j=1}^{N_h} u_j(t_n; \boldsymbol{\mu})\boldsymbol{\varphi}_j(\boldsymbol{x}_i)$. Here, $\{\boldsymbol{\varphi}_j(\boldsymbol{x})\}_{j=1}^{N_h}$ is a basis for V_h and $u_j(t; \boldsymbol{\mu})$ is the $j-$th coefficient of the solution of the linear system associated with (1).

To construct a reliable and robust dataset containing many possible combinations of environmental and operational conditions, we repeatedly solve (1) for different parameters. To overcome the computational burden associated with this step we resort to model order reduction techniques, see e.g., [21, 38], which seek to accurately approximate the underlying high-fidelity model by constructing a low-dimensional model by leveraging an offline–online decoupling. Indeed, the reduced model is built during an expensive offline phase, where a set of high-fidelity solutions are combined to fulfil a suitable orthogonality criterion. Then, in the online phase, for a new parameter, the reduced basis solutions are inexpensively obtained by solving a smaller linear system, i.e., the reduced problem. Finally, the solution is projected back to the original space. While the details of the reduced basis go beyond the scope of this work, we refer the reader to [5] and references therein for an in-depth description of how the reduced basis method can be used to solve the acoustic-elastic problem in frequency domain and how to reconstruct the time signal with numerical inverse Laplace transforms. Similarly, for the static problem, we refer the reader to [22, 13], for the details of the associated reduced model.

2.3. The chosen quantities of interest are the damage-sensitive features

In the SHM framework it is common to resort to damage-sensitive features, extracted from the raw displacements, to support the decision-making process, see e.g., [29, 48, 5]. From a mathematical standpoint, the desired feature function

$$\mathcal{F} = \mathcal{F}(\mathbf{u}_i(\boldsymbol{\mu})) : \mathbb{R}^{(N_t+1)\times d} \to \mathbb{R}^{Q\times d} \tag{4}$$

takes as input a discrete time signal (3) and outputs a set of Q d-dimensional features. In the context of guided-wave problems, feature extraction refers to the process of compressing raw sensor measurements, which are high-dimensional because of high sampling rates and possibly long time windows, i.e., both N_t and T are usually large, into low-dimensional vectors. Indeed, as the dimensionality of the training dataset grows, many state of the art machine learning algorithms, including anomaly detection models, become intractable. Dealing with a large number of features not only leads to poor generalization capabilities, but also to inefficient learning models with high computation costs. This phenomenon, known as curse of dimensionality, can be overcome by feature compression. As mentioned, the ideal features should be *damage-sensitive* and, at the same time, insensitive to the natural variation of the baseline operational and environmental conditions. Common choices for features for guided-waves approach can be found, e.g., in [29]. We follow the strategy presented in [5], where the authors use six features, i.e., the arrival time of the wave, the crest factor, the number of peaks and valleys as well as the minimum and the maximum amplitude in a fixed time window.

To further reduce the dimensionality of the output of interest after normalizing the features, we rely on principal component analysis (PCA), computed by a singular value decomposition to yield an orthonormal basis ordered by energy of variance. Indeed, the displacements along the d directions are correlated, leading to redundant features. The optimal number d_y of retained principal components, i.e., those with the highest variability, is determined by looking at the cumulative explained variance ratio as a function of the number of components. For the sake of notation, we let $\mathcal{F}$ include both the classic feature extraction and the subsequent PC compression, i.e., $\mathcal{F} = \mathcal{F}(\mathbf{u}_i(\boldsymbol{\mu})) : \mathbb{R}^{(N_t+1)\times d} \to \mathbb{R}^{d_y}$.

We remark that there exists alternative anomaly detection algorithms where the entire time signals can be used directly. For example, long short-term memory (LSTM) autoencoders are a type of recurrent neural networks (RNNs), successfully used in the context of speech recognition or text translation, see e.g., [19]. More generally, autoencoders are a type of neural networks, whose output is a reconstructed copy of the input [17]. The strength of autoencoders lies in the identification of a low-dimensional non-linear manifold where the input data lay on. This manifold can be used to reconstruct the full signal with few variables, called the latent variables. In particular, in the anomaly detection framework, the latent variables could play the role of the aforementioned features, with the main difference that the network would be purely data-driven, while the features are based on engineering knowledge. Despite this desirable property, it is less clear how autoencoders could be used for optimal sensor placement.

We finally observe that while signal compression is a fundamental step for the dynamic case, in the context of static loads, the formulation is greatly simplified. Indeed, since the problem is static, the vector of displacements (3) also becomes time-independent, i.e., $\mathbf{u}_i(\boldsymbol{\mu}) \in \mathbb{R}^d$ for $i = 1, \ldots, n_{\text{dof}}$. Moreover, the aforementioned compression process based on damage-sensitive feature extraction and PCA is not needed

when the quantities of interest are the discrete displacements. In this cases the feature map (4) is the identity map, i.e., $\mathcal{F} = \mathcal{F}(\mathbf{u}_i(\boldsymbol{\mu})) : \mathbb{R}^d \to \mathbb{R}^{d_y}$, with $d_y = d$.

To conclude, we remark that, in the online phase, the reduced problem has to be solved for n_μ random input parameters, possibly chosen from a fixed sampling strategy, e.g., Sobol sequence, Latin hypercube etc., to obtain the healthy dataset, i.e.,

$$\mathbf{Y}(\boldsymbol{\mu}_j) = [\mathcal{F}(\mathbf{u}_1(\boldsymbol{\mu}_j)), \dots, \mathcal{F}(\mathbf{u}_{n_{\text{dof}}}(\boldsymbol{\mu}_j))], \quad \text{for } j = 1, \dots, n_\mu, \tag{5}$$

where $\mathcal{F}$ is defined in (4).

3. Sparse GP Regression

The sparse GP regression has received increasing attention in the last decades thanks to its ability to overcome the computational limitation of a standard GP. Indeed, given the number of training samples n, the computational complexity of generating a GP model is $\mathcal{O}(n^3)$ and the associated storage requirement $\mathcal{O}(n^2)$, which becomes intractable for large data sets. The corresponding sparse methods instead rely on a small set of $m \ll n$ points to facilitate the information gain of the whole data set, thus allowing for a complexity reduction, i.e., $\mathcal{O}(nm^2)$. After a short introduction of GP regression in Section 3.1, we detail the properties and advantages of its sparse variation in Section 3.2. We discuss the formulation of variational inference of a sparse approximation in Section 3.3, which is of relevance to the method proposed in this paper.

3.1. A short review of GP regression models

A GP regression (GPR) model is a supervised machine learning approach, whose goal it is to construct a regression model to predict continuous quantities of interest given a set of observations. A GP is a set of random variables, any finite subset of which follows a Gaussian distribution. We observe that a GP is fully defined by its first and second moments. Without loss of generality, we take the mean function $m(\boldsymbol{x})$ to be zero. The covariance function $k(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta})$, also called the kernel function, is parametrized by a small set of hyperparameters $\boldsymbol{\theta}$, e.g., the variance of the kernel and the lengthscales of the input dimensions, thus incorporating some prior knowledge on the smoothness of the stochastic process and the similarity between data points.

Let $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ denote a training data set of d-dimensional inputs $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_n]^T$ and the corresponding real-valued realisation $\mathbf{y} = [y_1, \dots, y_n]^T$ of a latent function $f(\boldsymbol{x})$ corrupted by some Gaussian white noise ε, i.e.,

$$y_i = f(\mathbf{x}_i) + \varepsilon_i, \quad \text{where } \varepsilon_i \sim \mathcal{N}(0, \sigma_y^2),$$

where σ_y^2 is the variance of the noise. We assume a zero-mean GP prior over the latent function we are trying to model, i.e., $f(\boldsymbol{x}) \sim \text{GP}\left(0, k(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta})\right)$. Given the noisy dataset, this can be expressed by the marginal likelihood

$$p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta}) = \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{K}_{nn} + \sigma_y^2 \mathbf{I}_n),$$

where $\mathbf{K}_{nn}$ is the $n \times n$ covariance matrix with $[\mathbf{K}_{nn}]_{ij} = k(\mathbf{x}_i, \mathbf{x}_j; \boldsymbol{\theta})$, and $\mathbf{I}_n$ is the $n-$dimensional identity matrix. For the sake of convenience, we consider the variance of the noise σ_y^2 as an additional hyperparameter belonging to the set $\boldsymbol{\theta}$. The best performance of a GPR model, i.e., its ability to make accurate predictions, strongly depends on the hyperparameters. The optimal hyperparameters are estimated from the training data $\mathcal{D}$ by minimizing the negative log likelihood over the space of hyperparameters:

$$\boldsymbol{\theta}_{\text{opt}} = \arg\min_{\boldsymbol{\theta}} - \log\left[p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta})\right],$$

where

$$\log\left[p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta})\right] = \log\left[\mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{K}_{nn} + \sigma_y^2 \mathbf{I}_n)\right] = -\frac{1}{2}\mathbf{y}^T(\mathbf{K}_{nn} + \sigma_y^2 \mathbf{I}_n)^{-1}\mathbf{y} - \frac{1}{2}\log|\mathbf{K}_{nn}| - \frac{n}{2}\log 2\pi. \tag{6}$$

To predict the function values at p new test inputs $\mathbf{X}_* = [\mathbf{x}_{*1}, \dots, \mathbf{x}_{*p}]$, one assumes a joint GP prior of the latent function values for the training data $\mathbf{f}_n = [f(\mathbf{x}_1), \dots, f(\mathbf{x}_n)]^T$ and the unobserved function values $\mathbf{f}_* = [f(\mathbf{x}_{*1}), \dots, f(\mathbf{x}_{*p})]$, i.e.,

$$p(\mathbf{f}_n, \mathbf{f}_*) = \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{K}_{nn} & \mathbf{K}_{n*} \\ \mathbf{K}_{*n} & \mathbf{K}_{**} \end{bmatrix}\right).$$

Here, $\mathbf{K}_{*n} = \mathbf{K}_{n*}^T$ is the covariance matrix between the new inputs $\mathbf{X}_*$ and the training samples $\mathbf{X}$, i.e., $[\mathbf{K}_{*n}]_{ij} = k(\mathbf{x}_{*i}, \mathbf{x}_j; \boldsymbol{\theta}_{\text{opt}})$. Thus, the noise-free posterior distribution is obtained by conditioning the predictive targets $\mathbf{f}_*$ on the observations $\mathbf{y}$ and it has the following posterior mean and variance estimates

$$m_{\mathbf{y}}(\boldsymbol{x}_*) = \mathbf{K}_{*n}(\mathbf{K}_{nn} + \sigma_y^2 \mathbf{I}_n)^{-1}\mathbf{y},$$
$$k_{\mathbf{y}}(\boldsymbol{x}_*, \boldsymbol{x}_*) = k(\boldsymbol{x}_*, \boldsymbol{x}_*; \boldsymbol{\theta}_{\text{opt}}) - \mathbf{K}_{*n}(\mathbf{K}_{nn} + \sigma_y^2 \mathbf{I}_n)^{-1}\mathbf{K}_{n*}.$$

We finally remark that the performance of the predictive distribution peaks with a correct choice of the kernel function followed by an accurate estimation of the hyperparameters. Among the commonly used covariance functions, we consider the *automatic relevance determination squared exponential (ARD-SE)* kernel and the *ARD exponential (ARD-E)* kernel, i.e.,

$$k_{\text{ARD-SE}}(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta}) \coloneqq \sigma_f^2 \exp\left(-\frac{1}{2}r\right) \text{ and } k_{\text{ARD-E}}(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta}) \coloneqq \sigma_f^2 \exp\left(-\sqrt{r}\right), \text{ where } r = \sum_{j=1}^{d} \frac{(\boldsymbol{x}_j - \boldsymbol{x}'_j)^2}{\sigma_j^2}, \quad (7)$$

respectively. Above, $\boldsymbol{\theta} \coloneqq [\sigma_f^2, \sigma_1^2, \ldots, \sigma_d^2]$, where σ_f^2 is the output variance, which determines the average distance of the function away from its mean and σ_j^2 are the characteristic lengthscales for $j = 1, \ldots, d$. For more details on GPR models and kernel functions we refer the reader to [53, 52, 32].

3.2. Sparse GPR models

The non-parametric nature of GPR models makes them popular for the prediction of continuous functions. However, the training of a GPR model leads to a cubic scaling of the computational cost with the number of training samples. This complexity prevents GPRs to be used for big data sizes. To overcome this disadvantage, sparse approximations of GPR methods have been developed, providing an efficient training process that scales linearly with the number of training data. These methods rely on $m \ll n$ auxiliary latent variables, evaluated at some inputs $\mathbf{Z} \subset \mathbb{R}^m$, which are often referred to as the *inducing inputs*, to reduce the computational requirements to $\mathcal{O}(nm^2)$, thus making the sparse GPR competitive among machine learning methods for large data sets.

Following [39], we present an overview of sparse GPR methods. A crucial assumption in these models is that the training latent variables $\mathbf{f}_n$ and the test variables $\mathbf{f}_*$ are conditionally independent given the inducing variables $\mathbf{f}_m$, evaluated at the corresponding inducing points $\mathbf{Z} = [\mathbf{z}_1, \cdots, \mathbf{z}_m]^T$. This means that they can be expressed in two separate conditional distributions, i.e.,

$$p(\mathbf{f}_n, \mathbf{f}_*) \simeq \hat{p}(\mathbf{f}_n, \mathbf{f}_*) = \int \hat{p}(\mathbf{f}_*|\mathbf{f}_m)\hat{p}(\mathbf{f}_n|\mathbf{f}_m)p(\mathbf{f}_m)\mathrm{d}\mathbf{f}_m. \quad (8)$$

Different sparse approaches adopt different inducing conditional distribution approximations $\hat{p}(\mathbf{f}_*|\mathbf{f}_m)$ and $\hat{p}(\mathbf{f}_n|\mathbf{f}_m)$, while the inducing prior remains the same $p(\mathbf{f}_m) = \mathcal{N}(\mathbf{0}, \mathbf{K}_{mm})$. We mention here three algorithms, by chronological appearance, which build upon one another to achieve better approximations. First, the sparse greedy approximation to GPR proposed in [43] formulates the approximated joint prior (8) as follows

$$\hat{p}_1(\mathbf{f}_n, \mathbf{f}_*) = \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \hat{\mathbf{K}}_{nn} & \hat{\mathbf{K}}_{n*} \\ \hat{\mathbf{K}}_{*n} & \hat{\mathbf{K}}_{**} \end{bmatrix}\right).$$

Here $\hat{\mathbf{K}}_{ab} = \mathbf{K}_{am}\mathbf{K}_{mm}^{-1}\mathbf{K}_{mb}$ is the Nyström approximation of the true prior covariance $\mathbf{K}$, which leverages the information provided by the m inducing inputs. Intuitively, $\hat{\mathbf{K}}_{nn}$ and $\hat{\mathbf{K}}_{**}$ quantify how much information $\mathbf{f}_m$ provides about $\mathbf{f}_n$ and $\mathbf{f}_*$, respectively. The main drawback of this approach is that $\hat{\mathbf{K}}$ has only m degrees of freedom, i.e., the joint prior is degenerate, which results in overconfident predictions over a very limited family of functions. An alternative approximation is proposed in [11, 41], where the exact prior variance matrix $\mathbf{K}_{**}$ is employed instead of approximating it by the inducing variables:

$$\hat{p}_2(\mathbf{f}_n, \mathbf{f}_*) = \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \hat{\mathbf{K}}_{nn} & \hat{\mathbf{K}}_{n*} \\ \hat{\mathbf{K}}_{*n} & \mathbf{K}_{**} \end{bmatrix}\right). \quad (9)$$

In this way $\mathbf{f}_*$ retains its own prior variance, leading to more reasonable predictive uncertainties than those given by the previous model, even if the predictive means are identical. Further improvements on the joint

kernel approximation have been made in [45] with the Sparse Pseudo-input Gaussian processes (SPGPs) approximation, where a more sophisticated joint prior is employed:

$$\hat{p}_3(\mathbf{f}_n, \mathbf{f}_*) = \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \hat{\mathbf{K}}_{nn} + \mathrm{diag}[\mathbf{K}_{nn} - \hat{\mathbf{K}}_{nn}] & \hat{\mathbf{K}}_{n*} \\ \hat{\mathbf{K}}_{*n} & \mathbf{K}_{**} \end{bmatrix}\right).$$

Note that, as opposed to the previous two methods, the diagonal of $\hat{\mathbf{K}}_{nn}$ is corrected to be the exact one, thus imposing an additional independence assumption about the training conditional distribution $\mathbf{f}_n$ given $\mathbf{f}_m$.

A particular note should be made about the inducing variables, which, depending on the approach, can either be a subset of the training set $\mathbf{X}$ or arbitrary locations in the input space. The former selection strategy leads to a prohibitive combinatorial optimization, for which sub-optimal greedy-like solutions have been proposed to alleviate the computational complexity, see e.g., [44, 43, 41, 50]. Nevertheless, relaxing the constraint on the inducing variables as a subset of the training data can potentially lead to a better local optimizer, as the optimization continuous and the target space is now larger. However, we observe that, in both cases, reaching the global minimum is intractable and one can only expect to converge to a good local minimum. This limitation is common to the optimization of marginal likelihood functions, which are often non-convex with respect to the hyperparameters. A common trick to overcome this issue is to use multiple starting points for both the hyperparameters and the inducing inputs [9]. Ultimately, by considering the inducing inputs $\mathbf{Z}$ as extra kernel hyperparameters that parametrize the covariance, their optimal values can be obtained simultaneously by minimizing the negative log likelihood, i.e.,

$$(\mathbf{Z}_{\mathrm{opt}}, \boldsymbol{\theta}_{\mathrm{opt}}) = \arg\min_{\mathbf{Z},\boldsymbol{\theta}} -\log\left[\hat{p}(\mathbf{y}|\mathbf{X}, \mathbf{Z}, \boldsymbol{\theta})\right] = \arg\min_{\mathbf{Z},\boldsymbol{\theta}} -\log\left[\mathcal{N}(\mathbf{y}|\mathbf{0}, \hat{\mathbf{K}}_{nn}^{\mathrm{exact}} + \sigma_y^2\mathbf{I}_n)\right], \tag{10}$$

where, $\hat{\mathbf{K}}_{nn}^{\mathrm{exact}}$ is the top left submatrix of the chosen prior covariance $\hat{p}$, e.g., $\hat{p}_i$, $i = 1, 2, 3$.

We finally remark that the quantities in (10) are trained in $\mathcal{O}(nm^2)$, while the computational complexities for the predictive mean and variance are $\mathcal{O}(m)$ and $\mathcal{O}(m^2)$, respectively. We refer the reader to [39] and references therein for more details on the similarities and differences on various sparse methods for GPR.

3.3. Variational inference of sparse GPR

An alternative to the exact inference is variational inference, which is another popular method in statistics. Instead of minimizing the negative log likelihood (10), variational inference seeks to find an approximation of the true GP posterior $p(\mathbf{f}_*|\mathbf{y})$ among a given family of distributions. Observing the differences between the marginal log likelihoods (6) and (10), it is clear that, although the latter represents an exact inference, it is based on a modified prior and therefore a continuous optimisation of (10) with respect to $\mathbf{Z}$ will not converge to the true GP model. Variational inference instead seeks to overcome this by considering the inducing inputs as variational parameters, whose optimal values are to be estimated jointly with the hyperparameters.

In [50], a variational Gaussian distribution $q(\mathbf{f}_n)$ is chosen to approximate the exact posterior $p(\mathbf{f}_n|\mathbf{y})$ on the training function values $\mathbf{f}_n$, such that, with the assumption of conditional independence of $\mathbf{f}_n$ and $\mathbf{f}_*$ given the inducing variables $\mathbf{f}_m$, $p(\mathbf{f}_n|\mathbf{y})$ can be approximated by the variational posterior

$$q(\mathbf{f}_n) = \int p(\mathbf{f}_n|\mathbf{f}_m)q(\mathbf{f}_m)\mathrm{d}\mathbf{f}_m.$$

The optimized inducing variables and hyperparameters are thus obtained by minimizing the Kullback-Leibler (KL) divergence between the true posterior and the variational posterior. In [50], it is proposed to minimize the KL divergence of the augmented true posterior $p(\mathbf{f}_n, \mathbf{f}_m|\mathbf{y})$ and the augmented variational posterior $q(\mathbf{f}_n, \mathbf{f}_m) = p(\mathbf{f}_n|\mathbf{f}_m)q(\mathbf{f}_m)$, which is equivalent to maximize the variational lower bound

$$\mathcal{L}(\boldsymbol{Z}, \boldsymbol{\theta}) = \log\left[\mathcal{N}(\mathbf{0}|\hat{\mathbf{K}}_{nn} + \sigma_y^2\mathbf{I}_n)\right] - \frac{1}{2\sigma_y^2}\mathrm{Tr}(\mathbf{K}_{nn} - \hat{\mathbf{K}}_{nn}), \tag{11}$$

where the second term is the negative trace of $\mathbf{K}_{nn} - \hat{\mathbf{K}}_{nn}$ scaled with $(2\sigma_y^2)^{-1}$ and $\hat{\mathbf{K}}_{nn}$ is defined as in Section 3.2. The resulting $(\mathbf{Z}_{\mathrm{opt}}, \boldsymbol{\theta}_{\mathrm{opt}})$ can then be used to build the predictive distribution, which is given by

$$q(\mathbf{f}_*|\mathbf{y}) = \mathcal{N}\left(\hat{\mathbf{K}}_{*n}(\hat{\mathbf{K}}_{nn} + \sigma_y^2\mathbf{I}_n)^{-1}\mathbf{y}, \mathbf{K}_{**} - \hat{\mathbf{K}}_{*n}(\hat{\mathbf{K}}_{nn} + \sigma_y^2\mathbf{I}_n)^{-1}\hat{\mathbf{K}}_{n*}\right). \tag{12}$$

We note that this is exactly the one used in [11, 41], i.e., the approximation with a joint prior (9). In terms of the predictive distribution the two methods are the same. However, the variational method, with the extra regularization term, relies on a very different selection of the inducing inputs and the hyperparameters. As opposed to the exact inference defined in (10), this additional trace term acts as a regularizer of the log likelihood, i.e., it summarizes the total variance of the conditional prior $p(\mathbf{f}_n|\mathbf{f}_m)$ and, as such, it can be viewed as an accuracy indicator of predicting $\mathbf{f}_n$ given $\mathbf{f}_m$. Minimizing this term prompts a good overall estimation of the statistics of the training data. We further note that, in the variational inference setting, the inducing variables $\mathbf{Z}$ determine the flexibility of both $p(\mathbf{f}_n|\mathbf{f}_m)$ and $q(\mathbf{f}_m)$, and, hence, the posterior $q(\mathbf{f}_n|\mathbf{y})$.

Finally, we remark that GPy [18], a Gaussian process regression framework in Python, is used for the numerical implementation of the examples presented subsequently.

4. Variational approximation for systematic sensor placement

In this work, we seek to provide a systematic sensor placement strategy in the context of anomaly detection for SHM. We therefore assume that only synthetic data generated by undamaged configurations under different environmental and operational conditions are available, i.e., we have no information regarding the type and severity of the anomalies. This is a realistic assumption because it is likely that many different types of damages will occur in the life time of a structure. If, on one hand, simulating all possible damages and locations would not be computationally feasible, it would on the other hand not be reasonable to make the hypothesis that including in the training set only a few representative damage types will generalize well to other types and locations; instead, it is more likely that mis-detections would occur. On the contrary, anomaly detection strategies detect damages by characterizing the similarities among healthy samples and identify as damaged new samples with significantly different properties from the undamaged ones, see e.g., [36]. Mathematically, this corresponds to unsupervised or semi-supervised learning techniques as opposed to supervised algorithms, where a different class is assigned to every different type (or location) of damage. This poses a significant challenge in the context of sensor placement where one has to define a suitable cost function to be optimized with respect to the operational parameters, e.g., the candidate locations for the sensor placement, the available number of sensors and so on. Indeed, existing cost functions are usually formulated in terms of damage detectability, see e.g., [33], which is a well defined concept only when a finite number of damages is assumed.

To overcome this obstacle, we propose to train a sparse GPR model of the monitoring phenomena, represented here by a chosen quantity of interest, e.g., displacement, stress or a function of those, by means of variational inference. By fixing the number m of inducing variables as the number of sensors that the user wishes to place on the structure, we identify the sensor locations with the local optima $\mathbf{Z}_{\text{opt}}$, obtained from the optimization of the variational lower bound (11). Then, the learned sparse GP model can be used to predict the effect of having placed sensors at particular locations $\mathbf{Z}_{\text{opt}}$. We recall that the optimal inducing variables $\mathbf{Z}_{\text{opt}}$ are such that the KL divergence between $q(\mathbf{f}_n)$ and the true posterior $p(\mathbf{f}_n|\mathbf{y})$ is minimal. On one hand, $q(\mathbf{f}_n)$ being a good approximation of the exact posterior distribution $p(\mathbf{f}_n|\mathbf{y})$ implies that the inducing variables provide enough statistics for the observed data, i.e., the information in the training data $\mathbf{f}_n$ can be compressed well in $\mathbf{f}_m$. As a consequence, the sensor locations $\mathbf{Z}$ do not cluster on the boundaries of the input domain, thus preventing "waste" in the sensed information. On the other hand, minimizing the regularizing trace term in (11), which represents the total variance of the conditional prior distribution $p(\mathbf{f}_n|\mathbf{f}_\mathbf{m})$, ensures that the mean square error of reconstructing the training latent values $\mathbf{f}_n$ from the inducing variables $\mathbf{f}_m$ is small. Indeed, the variational approximation guarantees that the sparse predictive distribution is as close as possible to the exact predictive distribution. This minimizes the reconstruction error not only at the sensor locations, but in the rest of the domain too. Hence, leveraging the variational sparse GPR for optimal sensor placements provides a tool to maximize the statistical information gain on the whole computational domain when using a fixed number of sensors, while reducing the computational requirements when compared to a traditional GP kernel based method.

In this section we elaborate on how the numerical data obtained from healthy structures, as described in Section 2, and the variational sparse GPR presented in Section 3.3 are combined for optimal sensor placement. After introducing the notation, in Section 4.1 we present details on placing sensors through variational inference of sparse GP for one particular structure configuration, while in Section 4.2 we describe how we handle the parametric dependency characteristic of each configuration in the context of optimal

sensor placement. In Section 4.1, emphasis is given to an *ad-hoc* optimization setup which allows, on one hand, to constrain sensors to lie on a specific portion of the domain and, on the other hand, to deal with extremely large input data. Both requirements are indeed common in the context of SHM, where structures may be represented by billions of degrees of freedom and only certain locations might be admissible to place sensors. We conclude with a description on how this procedure can be used to provide information about the sensitivity of a fixed network of sensors in Section 4.3.

Let us consider a d-dimensional spacial domain $\Omega \subset \mathbb{R}^d$ with a suitable triangulation $\mathcal{T}_h$, where h represents the mesh size, leading to a total of n_{dof} mesh points $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_{n_{\text{dof}}}]$. Moreover, let $\Omega_\mu \subset \mathbb{R}^{d_\mu}$ be a d_μ-dimensional domain representing the space of natural variations of the parameters of an healthy structure, e.g., different operational loads, external excitements and material properties. For a given parameter combination $\boldsymbol{\mu} \in \Omega_\mu$, we assume that the inputs and outputs are mapped through a function f and that this process is corrupted by some Gaussian white noise $\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \sigma_y^2)$, i.e.,

$$\mathbf{y}_i(\boldsymbol{\mu}) = f(\mathbf{x}_i; \boldsymbol{\mu}) + \boldsymbol{\varepsilon}, \quad \text{for } i = 1, \dots, n_{\text{dof}}, \tag{13}$$

where $\mathbf{Y}(\boldsymbol{\mu}) = [\mathbf{y}_1(\boldsymbol{\mu}), \dots, \mathbf{y}_{n_{\text{dof}}}(\boldsymbol{\mu})]$ are the n_{dof} d_y-dimensional outputs of interest (5), e.g., displacements of an elastic structure or features extracted from time-dependent signals.

We point out that, in contrast to most of the cases where GPRs are employed, in this work, the training outputs $\mathbf{Y}(\boldsymbol{\mu})$ are not experimental, but simulated. As a direct consequence, for a given parameter $\boldsymbol{\mu}$, the map from inputs to outputs is known exactly, i.e., $f(\mathbf{x}_i; \boldsymbol{\mu})$ is a function of the discrete time-signals (3), as described in Section 2. Therefore, we do not focus on constructing a GPR model to predict the mean and variance of the outputs at new spatial locations. The novelty of our approach lies in the fact that the sparse GPR is adopted to place sensors systematically; placing a Gaussian prior on the input-output map, i.e., $f(\boldsymbol{x}) \sim \text{GP}(\mathbf{0}, k(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta}))$, allows us to employ the variational inference algorithm presented in Section 3.3 and thus to identify the location of sensors as the inducing inputs.

4.1. Constrained variational approximation

The variational learning of the hyperparameters and the inducing inputs are obtained by maximizing the variational lower bound (11), which is in general an unconstrained non-convex optimization problem. Indeed, even if we may have positivity constraints on some hyperparameters, e.g., the variance and lengthscales of the kernel function, the fact that we approximate the log value of those hyperparameters transforms the problem to an unconstrained optimization. While this may not be an issue for the aforementioned hyperparameters, which appear to be squared in the kernel functions (7), we do need to impose some locality constraints on the inducing points to prevent them to be outside the input domain, especially when this is non-convex. Moreover, in some particular scenarios in the framework of SHM, one has to consider that it may be only possible to place sensors on a portion of the asset, e.g., sensors should not be placed inside a solid 3D structure, or they could only be placed on the above-surface structure of an offshore wind turbine, or only on the core of a nuclear reactor, avoiding the reflector subdomain [2].

We consider sensor placement for a specific configuration, i.e., the input parameter $\boldsymbol{\mu}$ is fixed in (13). For succinctness, we neglect the parameter dependence in this part, i.e., $\mathbf{y}_i = \mathbf{y}_i(\boldsymbol{\mu})$. Let n_s be the number of sensors to be placed and $\Omega_s \subset \Omega$ the admissible domain for sensor locations. To overcome the issues related to unconstrained optimization mentioned above, the minimization of the negative variational lower bound (11) is modified as

$$(\mathbf{Z}_{\text{opt}}, \boldsymbol{\theta}_{\text{opt}}) = \underset{\mathbf{z} \in \Omega_s \forall \mathbf{z} \in \mathbf{Z},\, \boldsymbol{\theta}}{\arg\min} \; -\mathcal{L}(\mathbf{Z}, \boldsymbol{\theta}),$$

where $\mathbf{Z} = [\mathbf{z}_1, \cdots, \mathbf{z}_{n_s}]^T \subset \mathbb{R}^{d \times n_s}$ is the collection of the n_s sensor locations and each one of them is constrained to belong to Ω_s. The optimization problem can be solved by common gradient-based constrained optimization algorithms, see e.g., [51], when Ω_s is a continuous domain. However, in real-life engineering applications, due to the complexity of Ω_s, it may be cumbersome to specify its boundaries analytically and, in such cases, it is worth to replace Ω_s with a discrete counterpart comprising a finite number of admissible points $|\Omega_s| \gg n_s$. This clearly poses a challenge for gradient-based techniques, which are not very efficient in discrete settings. To deal with real-world problems, we propose to use the genetic algorithm (GA) in our process. The GA, a type of evolutionary optimization algorithm, takes inspiration in the natural selection and undergoes three main stages: selection, crossover, and mutation [12, 42]. Having received increasing attention in the recent decade in the field of discrete optimization, the GA has been used to address several

optimal sensor placement problems [20, 33]. We propose to combine the gradient-based optimization with the GA to form an EM-like algorithm. At first, we fix the inducing points $\mathbf{Z}$ and employ a gradient-based algorithm to optimize the hyperparameters $\boldsymbol{\theta}$. We then fix the hyperparameters and use the GA to find the optimal inducing points. We lastly iterate over these two steps until convergence is reached. This approach is summarized in Algorithm 1. For the sake of completeness, we observe that, in case of continuous admissible domains Ω_s, one can either choose to combine the two optimization steps mentioned above or to keep them separately by replacing the GA with another gradient-based constrained optimization to estimate the inducing points. The second approach is advantageous for continuous problems with a faster convergence. We finally remark that DEAP (Distributed Evolutionary Algorithms in Python) [16] is the framework used for the numerical implementation of the GA examples presented in this work.

Algorithm 1: Variational approximation for systematic sensor placement

Input: training dataset $\{\mathbf{X}, \mathbf{Y}\}$, admissible set Ω_s, and max iteration number $k_{\max}$
Output: optimal constrained inducing points and hyperparameters $(\mathbf{Z}_{\text{opt}}, \boldsymbol{\theta}_{\text{opt}})$
Initialization: set $k = 0$ and randomly initialize $\mathbf{Z}_k$ s.t. $\mathbf{z}_i \in \Omega_s$ for $i = 1, \ldots, n_s$
while *not converged and* $k < k_{max}$ **do**
  Compute the optimal hyperparameters $\boldsymbol{\theta}_{k+1} = \arg\min_{\boldsymbol{\theta}} -\mathcal{L}(\boldsymbol{Z}_k, \boldsymbol{\theta})$.
  Compute the optimal constrained locations $\boldsymbol{Z}_{k+1} = \arg\min_{\mathbf{z}\in\Omega_s \forall \mathbf{z}\in\mathbf{Z}} -\mathcal{L}(\boldsymbol{Z}, \boldsymbol{\theta}_{k+1})$
  Set $k = k + 1$
end
Set: $\mathbf{Z}_{\text{opt}} = \mathbf{Z}_k$, $\boldsymbol{\theta}_{\text{opt}} = \boldsymbol{\theta}_k$

4.2. Including parameter dependency in sensor placement

Let us reintroduce the parameter dependency and consider a set of n_μ parameters $\mathcal{D}_\mu = [\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_{n_\mu}]$, where $\boldsymbol{\mu}_j \in \Omega_\mu$ for $j = 1, \cdots, n_\mu$. Applying Algorithm 1 for all these parameters, we obtain a set of n_μ parameter-dependent inducing points $[\mathbf{Z}_{\text{opt}}(\boldsymbol{\mu}_1), \ldots, \mathbf{Z}_{\text{opt}}(\boldsymbol{\mu}_{n_\mu})]$, where $\mathbf{Z}_{\text{opt}}(\boldsymbol{\mu}_j)$ correspond to the n_s optimal locations for the specific parametric underlying system defined by $\boldsymbol{\mu}_j \in \Omega_\mu$. Having a continuous mapping from the inputs to the outputs and under the assumption that the parameters in Ω_μ only vary some accessory properties without altering the topology of the structure, it is reasonable to assume that each one of the n_s inducing points $\mathbf{Z}_{\text{opt}}(\boldsymbol{\mu}_j)$ lie in the neighborhood of the corresponding inducing point obtained for a different input parameter, i.e., $\mathbf{Z}_{\text{opt}}(\boldsymbol{\mu}_i)$ for $i \neq j$ and $i, j = 1, \ldots, n_\mu$. Therefore, to include the parametric dependency and summarize the information from this set of $n_s n_\mu$ into a set of n_s locations, we propose to employ the K-medoids algorithm to find n_s clusters and its corresponding centers.

Similar to K-mean algorithm, K-medoids is a clustering algorithm that breaks the data set into a user-defined number of groups and minimizes the distance of the center of each cluster and the points in it. The difference between these two clustering algorithms is that the K-means algorithm averages points within a cluster as the center, whereas K-medoids selects only data points as cluster centers. In comparison, K-medoids is more robust as the algorithm seeks to minimize the sum of dissimilarities of all points inside a cluster instead of the sum of squared Euclidean distances, as used in the K-means algorithm, which is sensitive to noise and outliers [4]. We point out that, in the numerical examples, the clustering step is carried out in Matlab [30] by employing the built-in function `kmedoids`. For more details on K-medoids algorithm, we refer the readers to [35, 4].

We summarize the algorithm for sensor placement that incorporates parameter variation of a solid structure in Algorithm 2. We notice that given different initial conditions, the K-medoids algorithm can lead to different clusters. The final decision can be made by either fixing the initial condition or by engineering experience across the resulting clusters.

4.3. A tool for sensor sensitivity

The technology proposed here can also be applied to answer a few related questions: (*i*) how many sensors are needed to achieve a prescribed precision? (*ii*) what is the expected sensitivity of a fixed sensor network? (*iii*) when a fixed network of n_s sensors already exists, given a budget of n_s^{extra} additional sensors, where

Algorithm 2: Parametrized variational approximation for systematic sensor placement

Input: parametric training dataset $\{\mathbf{X}, \mathbf{Y}(\boldsymbol{\mu}_j)\}_{j=1}^{n_\mu}$ and admissible set Ω_s
Output: optimal constrained sensor locations $\mathbf{Z}_{\text{opt}}$
for $j = 1, \cdots, n_\mu$ **do**
  Apply Algorithm 1 to data set $\{\mathbf{X}, \mathbf{Y}(\boldsymbol{\mu}_j)\}$ to get n_s inducing inputs $\mathbf{Z}_{\text{opt}}(\boldsymbol{\mu}_j)$ constrained to Ω_s
end
Apply K-medoids algorithm to the $n_s n_\mu$ inducing inputs $[\mathbf{Z}_{\text{opt}}(\boldsymbol{\mu}_1), \ldots, \mathbf{Z}_{\text{opt}}(\boldsymbol{\mu}_{n_\mu})]$ to get n_s clusters
Set: $\boldsymbol{Z}_{\text{opt}}$ = cluster centers

should these be placed to achieve optimal coverage? Properly addressing these queries is of great importance in the maintenance of real-life engineering problems.

The first point refers to the need of defining a suitable measure to quantify the quality of the locations, whether they are obtained with the proposed variational approach or already placed on the monitored structure. A straightforward choice is to compute the reconstruction of the quantity of interest, i.e., $m^q_{\mathbf{Y}(\boldsymbol{\mu}_j)}(\mathbf{x}_i)$ at all training points $\mathbf{x}_i \in \mathbf{X}$, for $i = 1, \ldots, n_{\text{dof}}$. Here $m^q_{\mathbf{Y}(\boldsymbol{\mu}_j)}(\mathbf{x}_i)$ is the mean of the posterior distribution (12) of the sparse model based on the variational parameters, i.e., outcome of Algorithm 2. Hence, the relative reconstruction error of the quantity of interest at *unsensed* locations can be used as an indicator of the sensor sensitivity. On one hand this quantity grows as we move away from the sensors and, on the other hand, increasing the number n_s of sensors is expected to improve the global coverage. Moreover, we define the average relative reconstruction error over the n_μ samples as

$$R = \sum_{j=1}^{n_\mu} \frac{1}{n_\mu} \frac{\|\mathbf{Y}(\boldsymbol{\mu}_j) - m^q_{\mathbf{Y}(\boldsymbol{\mu}_j)}(\mathbf{X})\|}{\|\mathbf{Y}(\boldsymbol{\mu}_j)\|}, \tag{14}$$

where $\mathbf{Y}(\boldsymbol{\mu}_j)$ is the simulated quantity of interest (5). A low R value is an indicator of a good global placement which takes the parametric dependency of the structure into account. An additional indicator to quantify the quality of sensor placement is the point-wise relative variance reduction, defined as

$$V_i = \frac{\mathbf{K}_{im}\mathbf{K}^{-1}_{mm}\mathbf{K}_{mi}}{\mathbf{K}_{ii}}, \quad \text{for } i = 1, \ldots, n_{\text{dof}}, \tag{15}$$

where $\mathbf{K}$ is the kernel matrix with optimized hyperparameters defined in Section 3. This quantity expresses how much variance reduction can be achieved by including the chosen sensor locations. A relative variance reduction close to one indicates that the inducing variables alone can reproduce the full GP prediction well.

Finally, we note that in the variational inference framework of the proposed approach, it is possible to jointly optimize some inducing inputs and keep the already existing sensor locations fixed. Thus, the strategy presented in this work can be efficiently implemented to systematically place additional sensors while accounting for the already existing structural coverage.

5. Numerical results

In Sections 5.1, 5.2, and 5.3, we provide examples of sensor placement in two and three dimensions for which we use the methodology presented in Section 4. A wave-based monitoring strategy is employed for the 2D and 3D examples given in Sections 5.1 and 5.2, respectively. Here, we resort to the mean reconstruction error and the relative variance reduction to test the quality of the sensor locations. Section 5.3, instead, presents a real-life engineering example, for which a static monitoring approach is used. Taking into account the complexity of the geometry and the large number of degrees of freedom, tests to assess the good quality of the placement are performed by looking at the achieved accuracy in detecting damages. The synthetic databases used in the training phase are constructed following the procedure given in Section 2.

5.1. Two-dimensional examples for the guided-wave problem

The examples in this section follow the wave-based monitoring approach, for which we train a variational sparse GP model with compressed signals. We consider the same governing problem (1) for three different

geometries shown in Figure 1 and we refer to these problems as Problems $1a$, $1b$, $1c$, whose domains will be identified by Ω_a, Ω_b, and Ω_c, respectively.

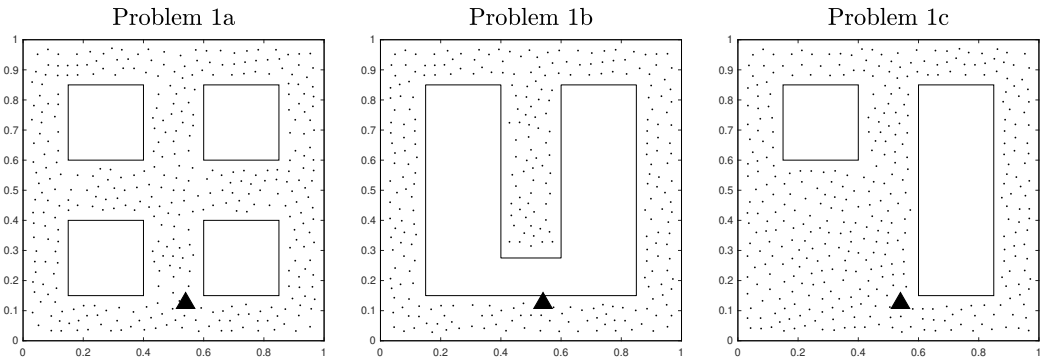


Figure 1: 2D examples with different geometries: Problem $1a$ relies on 360 training inputs (*small black dots*), corresponding to the vertices of a coarse mesh over the domain, while Problems $1b$ and $1c$ have 286 and 375 training points, respectively. The location of the center of the active source is the same for three geometries and corresponds to $\bar{S} = [0.54, 0.125]$ (*black triangle*).

For each problem, we consider zero initial conditions for both the displacement and the velocity and prescribe free slip boundary conditions, i.e.,

$$\begin{cases} \boldsymbol{u} \cdot \boldsymbol{n} = \boldsymbol{0} \\ (\boldsymbol{\sigma} \cdot \boldsymbol{n}) \cdot \boldsymbol{\tau} = \boldsymbol{g}_N \end{cases} \text{on } \partial\Omega,$$

where $\boldsymbol{\tau}$ is the tangential vector to $\partial\Omega$ and $\boldsymbol{g}_N = \boldsymbol{0}$ for simplicity. The high fidelity numerical solutions of (1) are computed using the FE approximation by $\mathbb{P}_1$ elements over a domain discretized in tetrahedral cells with a total of $N_h = 30'912$ degrees of freedom, while for the RB solver we rely on 267 basis for Problem $1a$. Similar order of magnitudes of these parameters are used for the other two problems: $N_h = 31'200$ and 284 basis for Problem $1b$ and $N_h = 26'072$ and 306 basis for Problem $1c$. For the discretization in time, we consider $N_t = 20'000$ and $T = 20$ for the three problems. The natural variations are described by three parameters, i.e.,

$$\boldsymbol{\mu} = [E, \nu, k] \in \Omega_\mu = [0.999, 1.001] \times [0.329, 0.331] \times [1.9, 2.1] \subset \mathbb{R}^3, \tag{16}$$

where E is the Young's Modulus, ν the Poisson's ratio which determines the Lamé constants (2) and k is a parameter of the active source function $s(\boldsymbol{x}, t; \boldsymbol{\mu})$, defined as follows

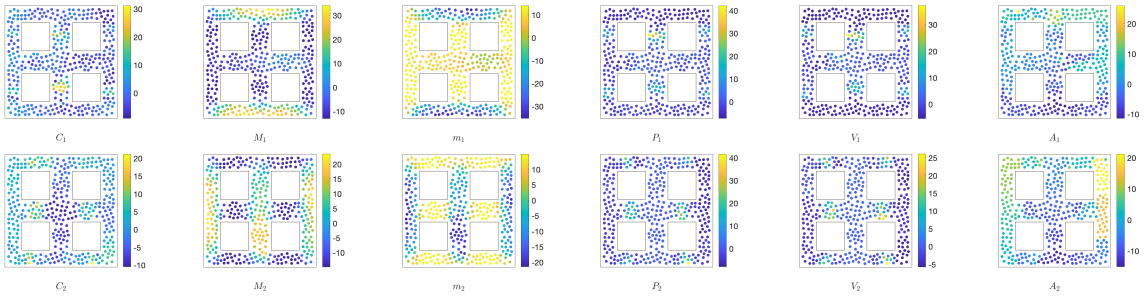
$$s(\boldsymbol{x}, t; \boldsymbol{\mu}) = \frac{\exp\left\{-\sum_{i=1}^{d} \frac{(\boldsymbol{x}_i - \bar{\mu}_i)^2}{2\bar{\sigma}_i^2}\right\}}{2\pi\bar{\sigma}^d} \, k_s \sin(k\pi t)\, te^{-t}. \tag{17}$$

Here, $\bar{\sigma} = 0.01$ represents the width of a Gaussian centered at $\bar{S} = [0.55, 0.125]$ with fixed amplitude coefficient $k_s = 100$. The parameter k represents the number of cycles before attenuation of the source impulse. For each problem we consider $n_\mu = 100$ samples and, to obtain a well balanced dataset, we sample from a Sobol's sequence [23], i.e., a base-2 digit sequence which provides a successively finer uniform partition of the intervals Ω_μ. We note that the density and damping coefficients are fixed, i.e., $\rho = 1$, $\eta = 0.1$, respectively.

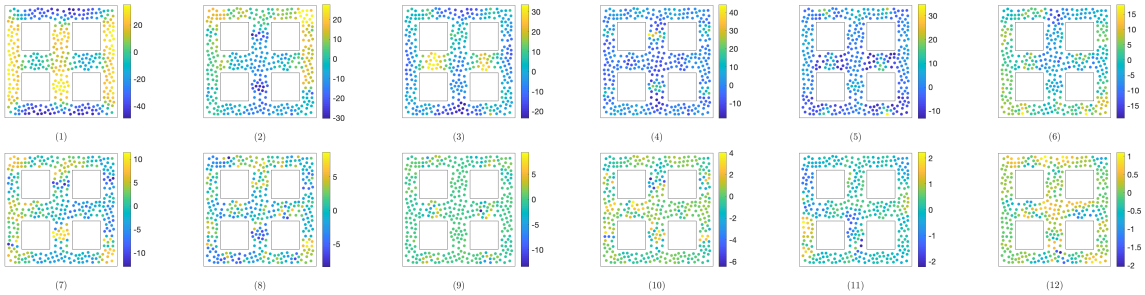
The training points $\mathbf{X} \subset \mathbb{R}^{n_{\text{dof}}} \subset \Omega_i$ with $i = a, b, c$ are obtained by fixing the same size of a coarse mesh for the three problems, thus recovering $n_{\text{dof}} = 360$, $n_{\text{dof}} = 286$, and $n_{\text{dof}} = 375$ mesh points, for Problems $1a, 1b$, and $1c$ respectively[1]. We observe that the mesh points on the boundary are not included in the training set. This correspond to a practical choice due to the free-slip boundary conditions, for which at least one of the the two displacement directions will be identically zero on each boundary edge. For each geometry we consider $d_y = 3$ quantities of interest (5) to train the variational sparse GP, i.e., the first

[1]We note that the n_{dof} degrees of freedom refer to the number of training points for the sensor placement strategy and they are independent from the N_h degrees of freedom used in the numerical simulations in Section 2.2. In general, $n_{\text{dof}} \ll N_h$.

three principal components of the $Q = 12$ features extracted from the discrete time-dependent displacement signals (3), obtained for $\boldsymbol{\mu}_1, \dots \boldsymbol{\mu}_{n_\mu}$. We note that for Problems 1a, 1b, and 1c, the first three principal components account for more than 80% of the variability. By way of example, Figure 2 shows the normalized features over the n_μ samples and the corresponding principal components for Problem 1a with $\boldsymbol{\mu} = [1, 0.33, 2]$. Normalization is performed by features, i.e., the means $\bar{m}_1, \dots, \bar{m}_Q$ and variances $\bar{\sigma}_1, \dots, \bar{\sigma}_Q$ are computed for each one of the Q features over all training points (e.g., $n_{\text{dof}} = 360$ for Problem 1a) and all simulations obtained for n_μ input parameters.



(a) Normalized features.



(b) Principal components.

Figure 2: Example of normalized features extracted from the solution obtained by solving the acoustic-elastic problem on the geometry 1a with $\boldsymbol{\mu}_1 = [1, 0.33, 2]$ (a). The first and second row show the 6 features related to the displacement along the x and y directions, respectively for a total of $Q = 12$ features. The Q corresponding principal components are shown in (b). The first three principal components account for 60.5%, 13.3%, and 11.5% of the variability, respectively for a total of more than 85%. Similar values are obtained for all the other samples and, for the other two geometries, i.e., Problems 1b and 1c, the importance of the three components is more balanced. The mean and standard deviation used for the normalization are based on the features extracted from $n_\mu = 100$ samples, obtained using the first 100 parameters of a Sobol sequence based on Ω_μ.

In terms of setup for the GPR, we note that for all the three examples, we use the ARD-Exponential kernel (7), which provide the best performance on the training set with respect to other popular choices, the Squared Exponential, Matérn-32 ad Matérn-52, both ARD and not.

By applying the sensor placement methodology described in Section 4 for $\{\mathbf{X}, \mathbf{Y}(\boldsymbol{\mu}_j)\}_{j=1}^{n_\mu}$, we obtain the systematic placement of sensors shown in Figure 3. For each geometry, the plots overlay the locations of the $n_s = 4, 9, 16, 25$ inducing points obtained by applying Algorithm 1 n_μ times over the admissible domains Ω_a, Ω_b, and Ω_c, i.e., a total of $n_s n_\mu$ inducing inputs, sometimes overlapping, is shown. The sets of inducing points are compared with the corresponding centroids, obtained by applying Algorithm 2, and, as an example, the inducing points obtained by applying Algorithm 1 for the first Sobol's parameter $\boldsymbol{\mu}_1 = [1, 0.33, 2]$ are also shown. While for larger numbers of inducing points, clusters appear to be more visible, for smaller n_s, the location of the $n_s n_\mu$ inducing inputs shows more variability. This can be explained by the fact that the optimal inducing inputs are optimized to reconstruct different quantities of interests, which depend on the input parameter $\boldsymbol{\mu}_j$. However, one also have to consider that, when trying to reconstruct a non-trivial quantity of interest over a complex structure with only few n_s points, the sparse model might get stuck in a

local minimum without reaching convergence. For example, the inducing points obtained for $\boldsymbol{\mu}_1$ for Problem $1a$ and $n_s = 9$ are not very well distributed over the entire domain. Nevertheless, the centroids seem to be a good summary of the entire underlying phenomena. Indeed, as shown in Figure 4, the optimal centroids obtained by clustering the results over the first $n_\mu = 10$ or the entire parameter domain, i.e., over $n_\mu = 100$ sample, are almost always indistinguishable. We note that purple stars in Figure 4 correspond to the same centroids shown in Figure 3, i.e., obtained by averaging the results of $n_\mu = 100$ samples.

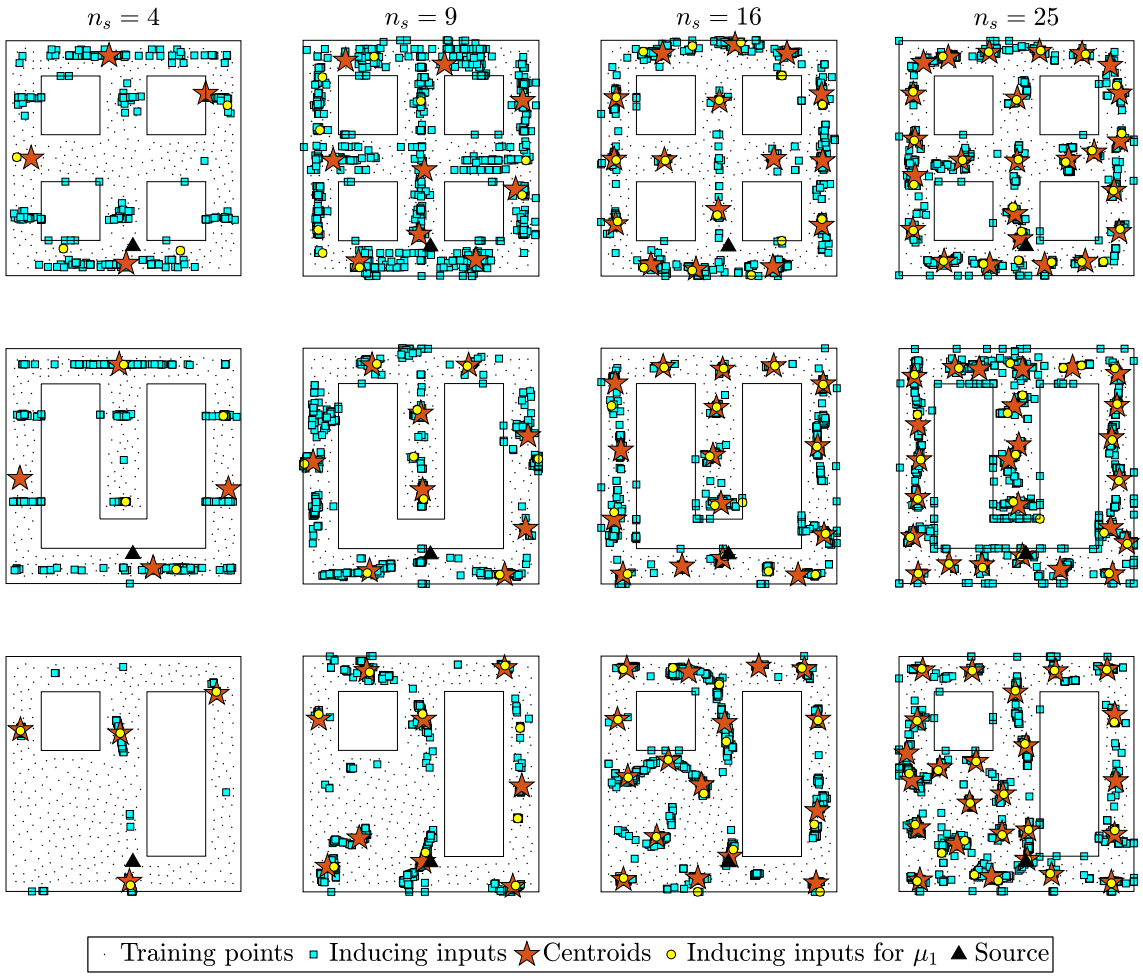


Figure 3: Comparison of the location of inducing points obtained by applying Algorithm 1 for $n_\mu = 100$ samples (*cyan squares*) and the corresponding n_s centroids obtained with Algorithm 2 (*red stars*). The inducing points obtained for one particular sample, i.e., $\boldsymbol{\mu} = [1, 0.33, 2]$, are also shown (*yellow circles*). Each row shows a different geometry while each column shows a fixed number n_s of inducing points, which increases from left to right, i.e., $n_s = 4, 9, 16, 25$.

As mentioned in Section 4.3, two ways to quantify the quality of the sensor placement outcome are by means of the reconstruction error and the variance reduction. Figure 5 shows the point-wise mean reconstruction of the first sample for Problems $1a$, i.e., $m^q_{\mathbf{Y}(\boldsymbol{\mu}_1)}(\mathbf{x}_i)$ with $\mathbf{x}_i \in \mathbf{X}$. We observe that as n_s increases, the different characteristics of the three principal components become visible in the reconstruction. We also note that reconstruction accuracy achieved for the first principal component $\mathbf{Y}_1$ is higher than the one for the other two. Indeed, the highest variability of the first principal component correspond to a less noisy field, simpler to be reconstructed by means of GPR. We remark that similar results are obtained for Problems $1b$ and $1c$. Figure 6 shows, for the three problems, the mean reconstruction error over the n_μ

Figure 4: Comparison of the centroids obtained with Algorithm 2 for different number of samples n_μ, namely $n_\mu = 10, 40, 70$ and 100. Each row shows a different geometry while each column shows a fixed number n_s of inducing points, which increases from left to right, i.e., $n_s = 4, 9, 16, 25$.

samples, defined in (14), for the three quantities of interests as a function of the number n_s of inducing points. These errors are compared to those obtained by reconstructing the principal components using the centroids as fixed variational hyperparameters in a new sparse GPR model. We observe that the difference between these two is minimal, which implies that the centroids are good approximations of the inducing points for sensor placement. Finally, Figure 7 shows the relative variance reduction (15), averaged over the n_μ samples. A variance reduction above 0.7 almost everywhere even for $n_s = 4$ is an indication of good sensor placement.

To conclude, Figure 8 compares the position of the centroids obtained with Algorithm 2 with the centroids obtained by applying the K-medoids algorithm to the training points $\mathbf{X}$ directly. This strategy is chosen as a proxy to place points *equidistantly* over a complex domain. Although this naive strategy may seem to give almost as good results as the laborious methodology followed to obtain the variational centroids, as shown in Figure 9, placing sensors without including physical information does not yield a good result. Indeed, the mean reconstruction accuracy obtained by training a new variational sparse GP model with fixed inducing inputs as the centroids obtained by K-medoids on the training points is not as good as the one obtained with variational centroids.

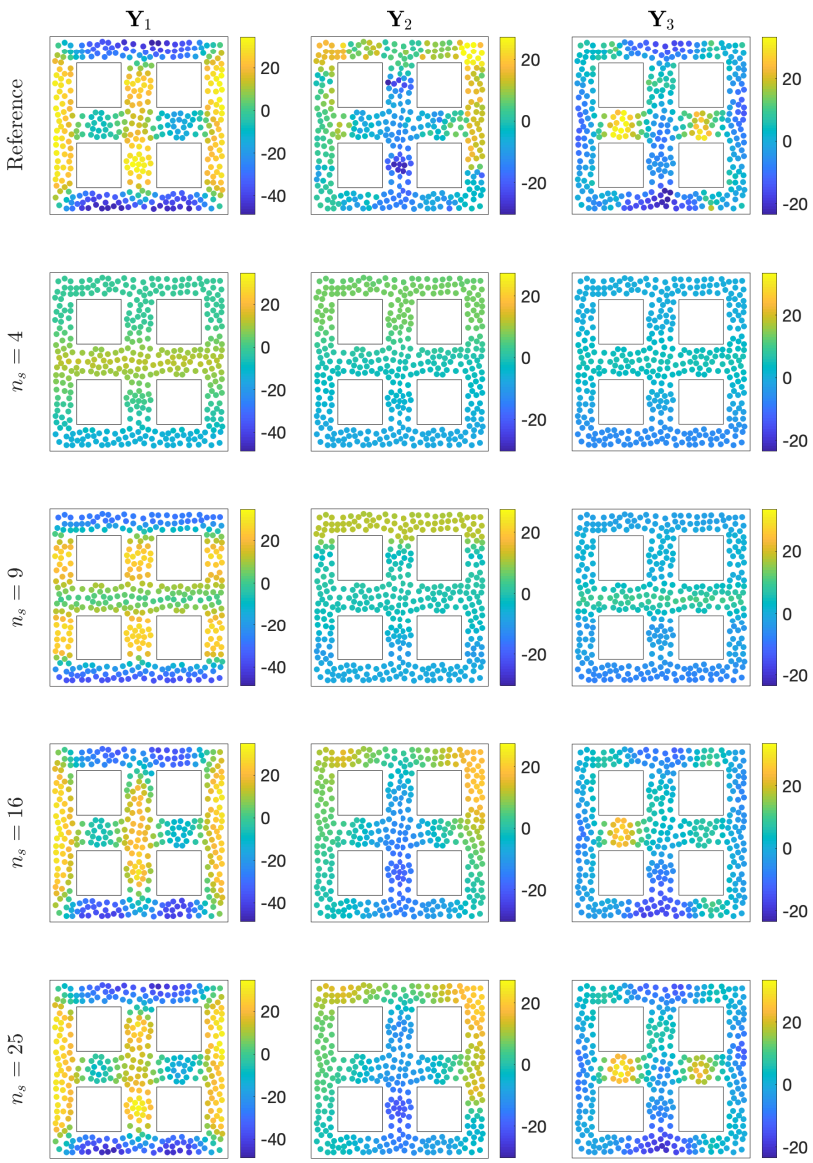


Figure 5: Comparison of the first three principal components obtained for Problem 1a either by extracting the features from the time signals and then performing PCA (*first row*) or by sparse GP reconstruction using $n_s = 4, 9, 16$ or 25 inducing points (*second to fifth rows*). As the number of inducing points increases, the output of interests can be better reconstructed. The *reference* principal components correspond to the results obtained for $\boldsymbol{\mu}_1 = [1, 0.33, 2]$. The color scale is the same for the reference and the corresponding reconstructions.

5.2. A three-dimensional example for the guided-wave problem

The sensor placement strategy following the guided-wave monitoring approach can be extended to 3D problems. Let us consider the geometry of a T-beam as shown in Figure 10. We consider the acoustic-elastic model (1) with zero initial conditions and homogeneous Dirichlet boundary conditions imposed on the surface $z = 0$ together with zero traction on the remaining surfaces. We compute the high fidelity solutions using the FE approximation by $\mathbb{P}_1$ elements over a fine mesh with $N_h = 262'863$ degrees of freedom and for the low fidelity model we use 505 basis. For the time discretization, we set $N_t = 10'000$ and $T = 10$. We consider the same parameter space (16) as for the 2D problem, where k is the free parameter of the the active source function (17), centered at $\bar{S} = [0.7, 1, 2]$. The training dataset corresponds to $n_{\text{dof}} = 4688$ input points of a coarse mesh restricted to the Neumann surfaces and $d_y = 4$ output of interests, i.e., the first four principal

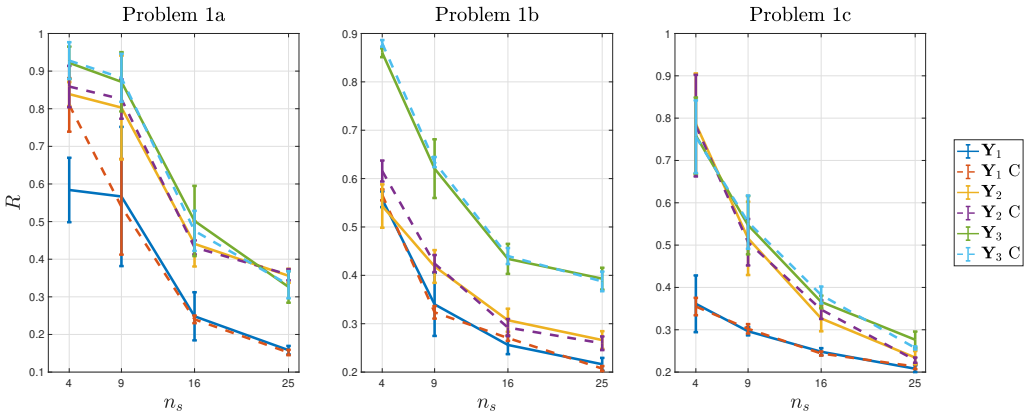


Figure 6: Mean reconstruction errors with error-bars with respect to the number n_s of inducing points for the first, second and third principal components (*solid lines*) used to train the variational sparse GP model. The corresponding mean reconstruction error, obtained by training a new variational sparse GP model with fixed inducing inputs corresponding to the centroids, is also shown (*dashed lines*). Each plot shows the result for one of the three geometries.

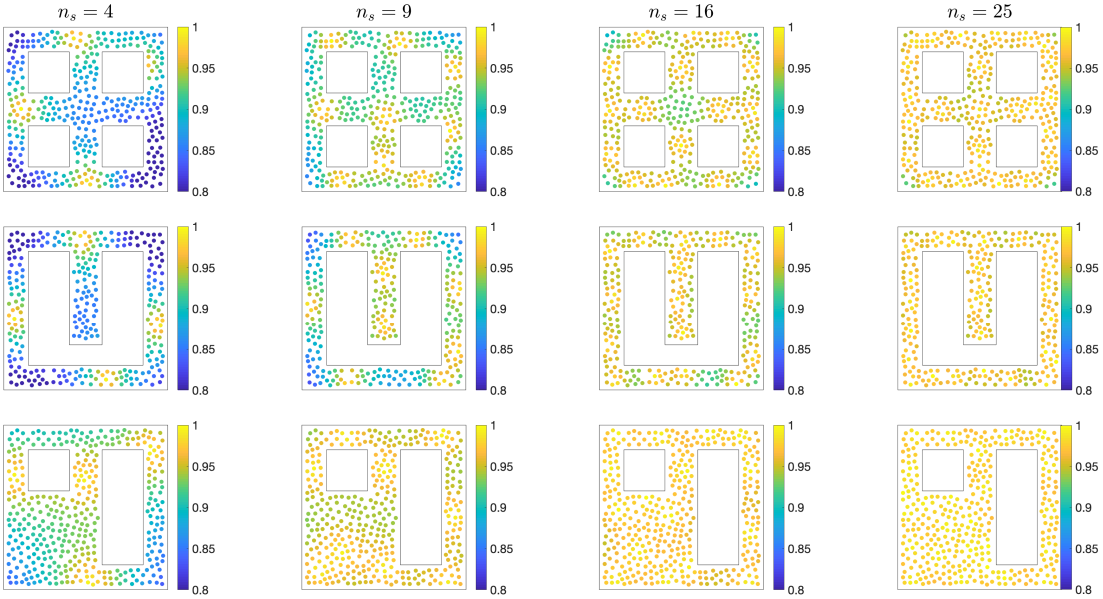


Figure 7: Relative variance reduction (15) obtained using n_s centroids and averaged over n_μ samples. Each row shows a different geometry while each column corresponds to a fixed number n_s of inducing points, which increases from left to right, i.e., $n_s = 4, 9, 16, 25$. The color scale is the same for all the plots.

components of the normalized $Q = 18$ features, extracted from the discrete time signals, as described in Section 2.3. We note that the union of the first four principal component accounts for more than 90% of the total variability for all samples. By way of example, the first two components obtained for $\boldsymbol{\mu}_1 = [1, 0.33, 2]$ are shown in the first row of Figure 12. After running Algorithm 2 for $n_\mu = 10$ Sobol's parameters, we obtain the inducing points and the centroids of the K-medoid clusters shown in Figure 11. Figure 12 also shows the mean reconstruction of the first two output of interest $m^q_{\mathbf{Y}_j(\boldsymbol{\mu}_1)(\mathbf{X})}$, for $j = 1, 2$, over the training set $\mathbf{X}$ for a fixed parameter $\boldsymbol{\mu}_1$ and increasing number of sensors, i.e., $n_s = 4, 16, 36$. As expected, the different characteristics of the output of interest become more visible in the predictions as the number of sensors increases. Finally, the relative variance reduction (15), with respect to the centroids and averaged over n_μ samples, is shown in Figure 13 for all training points. An overall relative reduction above 92% is achieved already for $n_s = 4$ sensors.

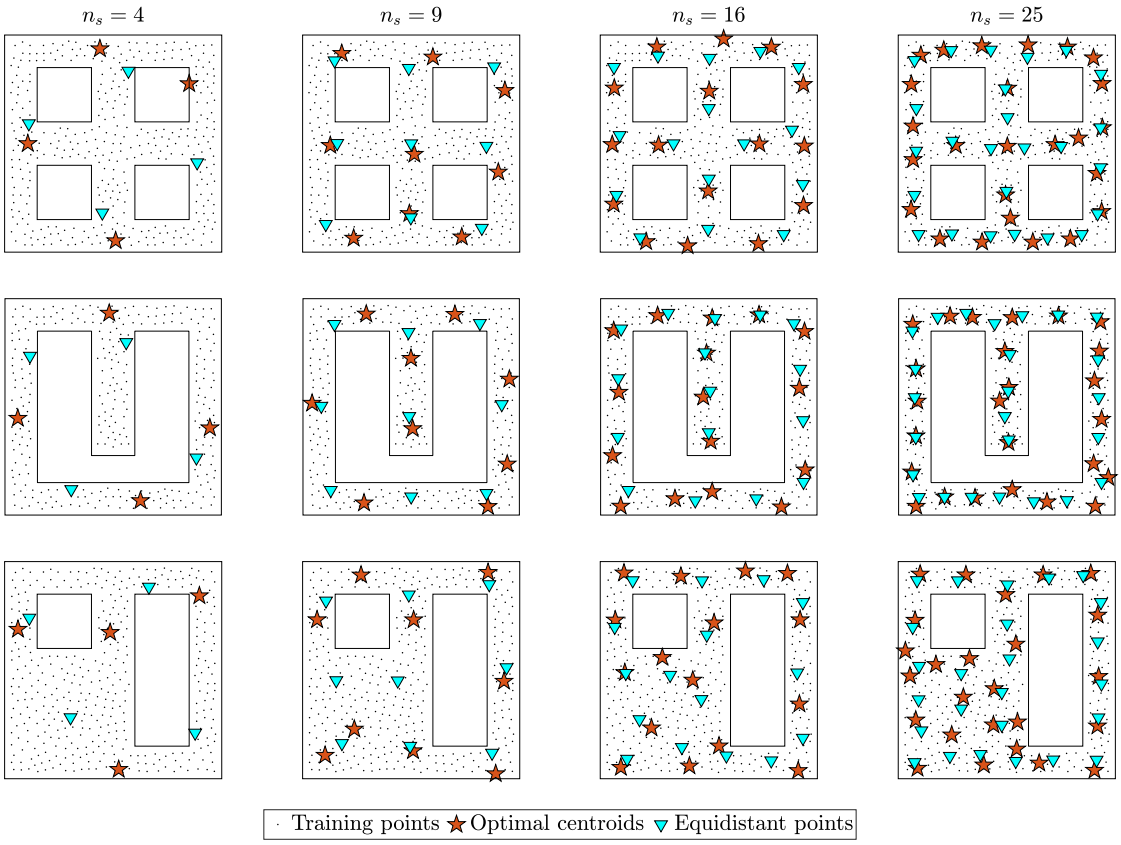


Figure 8: Comparison of centroids obtained using Algorithm 2 (*red stars*) and the naive clustering, referred to as equidistant points (*cyan down-facing triangles*). Each row shows a different geometry while each column shows a fixed number n_s of inducing points, which increases from left to right, i.e., $n_s = 4, 9, 16, 25$.

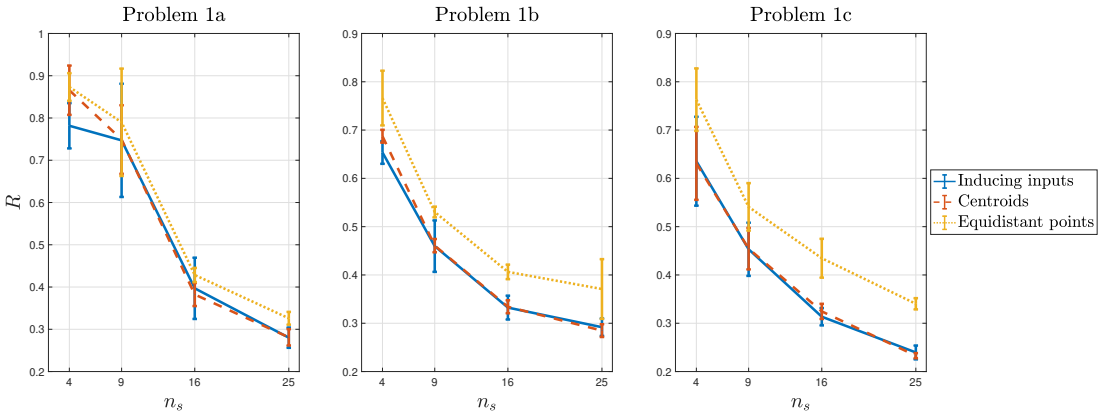


Figure 9: Mean reconstruction errors with error-bars with respect to the number n_s of inducing points for the three quantity of interest jointly (*solid line*) used to train the variational sparse GP model. The corresponding mean reconstruction error, obtained by training a variational sparse GP model with fixed inducing inputs corresponding to the centroids is also shown (*dashed line*) together with the one where the fixed inducing inputs are the naive centroids (*dotted line*). Each plot shows the result for one of the three geometries.

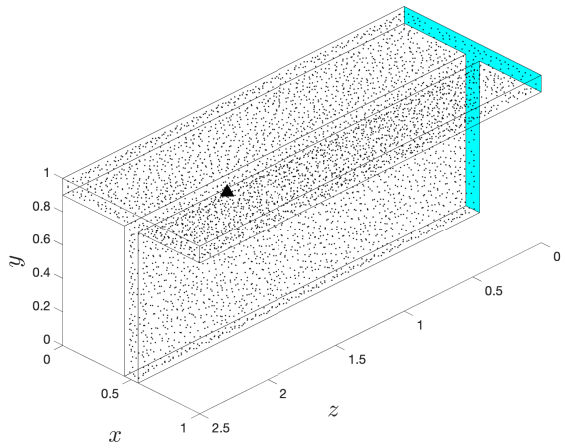


Figure 10: 3D geometry of a T-beam with 4688 training points (*small black dots*), corresponding to the vertices of a coarse mesh over the domain. The location of the center of the active source corresponds to $\bar{S} = [0.7, 1, 2]$ (*black triangle*). The Dirichlet boundary corresponds to the surface at $z = 0$ (*cyan filled surface*).

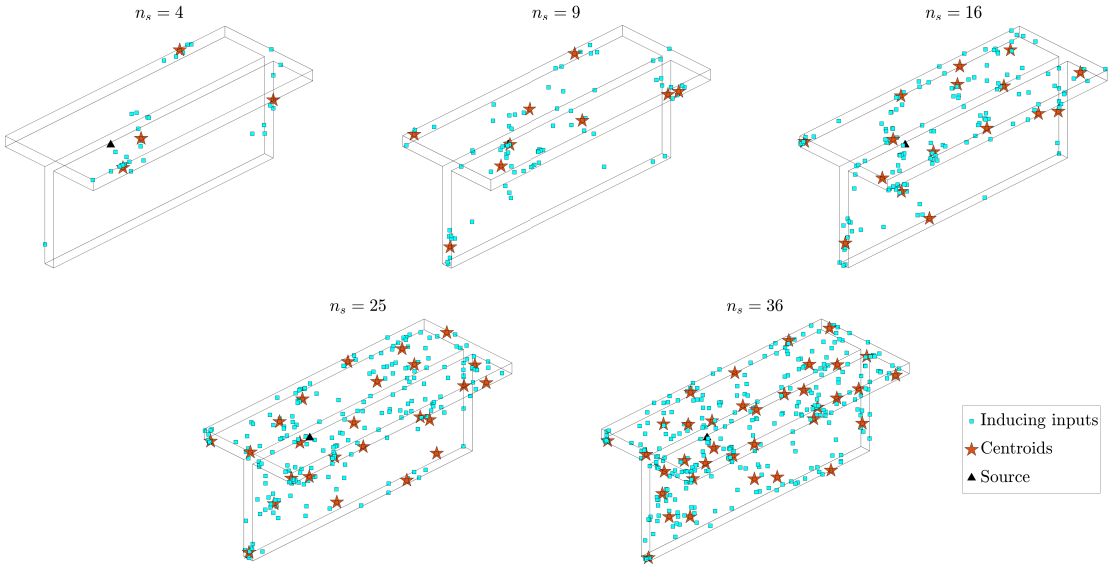


Figure 11: Comparison of the location of inducing points obtained by applying Algorithm 1 for $n_\mu = 10$ samples (*cyan squares*) and the corresponding n_s centroids obtained with Algorithm 2 (*red stars*). Each plot shows a different fixed number n_s of inducing points, i.e., $n_s = 4, 9, 16, 25, 36$.

5.3. Application to a realistic geometry of an offshore jacket

We now consider a real-life engineering example of an offshore jacket, consisting of 192 components, as shown in Figure 14. The bottom of the jacket is fixed on the ground and other boundaries are assumed to be free. We introduce two parameters, $\mu_x, \mu_y \in \Omega_\mu = [0.1, 1]$ kPa, representing the surface wind loads on the 64 components in the dark box in Figure 14 in the x and y directions, respectively. We assume the jacket to be linear elastic with Young's modulus $E = 200$ GPa and Poisson's ration $\nu = 0.3$. As mentioned in Section 2.3, the displacements under different load combinations are chosen as quantity of interest. The degrees of freedom of the full model exceed four million in the original finite element model which is solved by the SCRBE solver from Akselos [1]. To further accelerate the process, the degrees of freedom can be drastically reduced by taking a random subset of points within each component as representatives of that component.

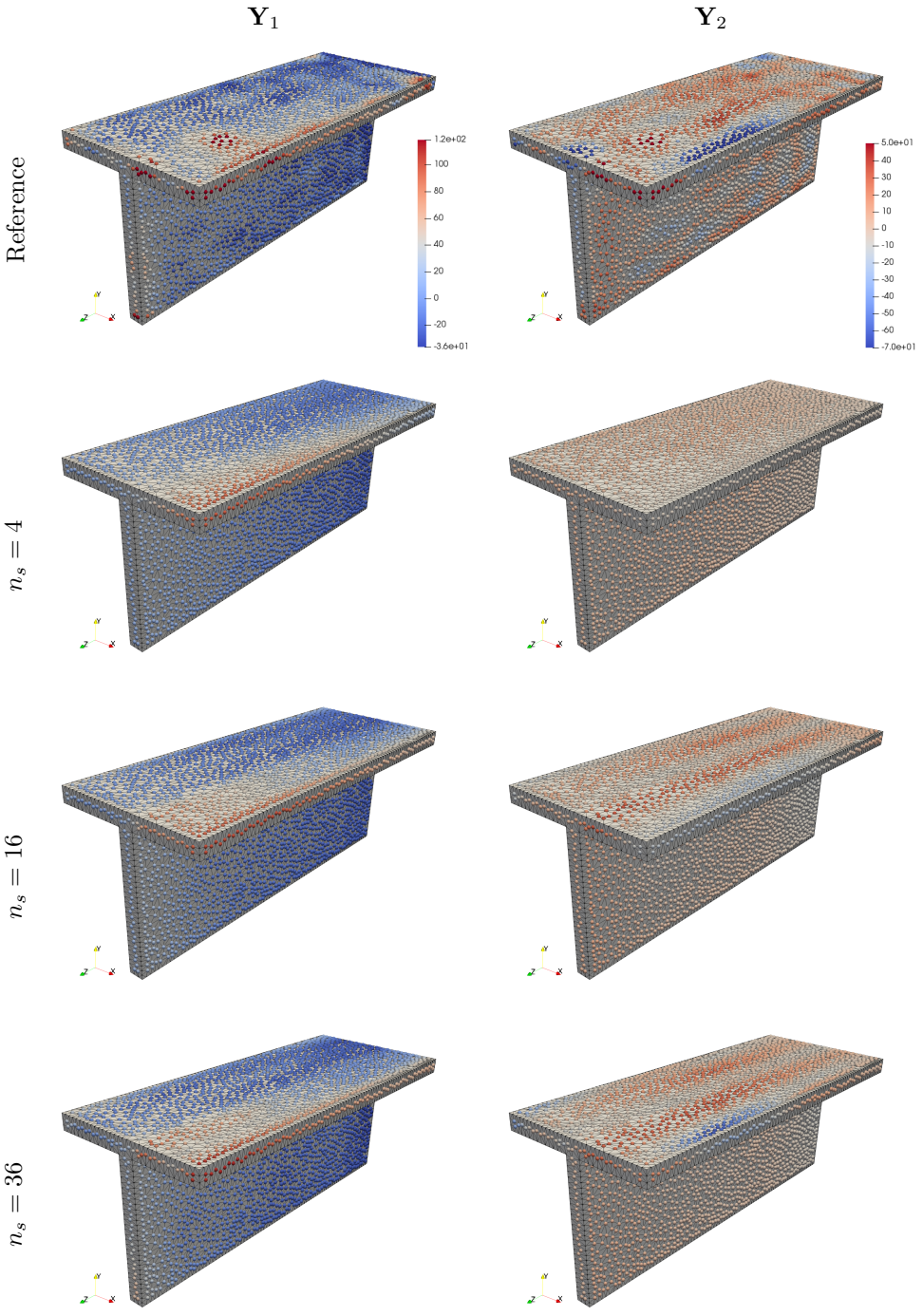


Figure 12: Comparison of the first two principal components obtained by extracting the features from the time signals and then performing PCA (*first row*) or by sparse GP reconstruction using $n_s = 4, 16$, or 36 inducing points (*second to fourth rows*). As the number of inducing points increases, the output of interests can be better reconstructed. The *reference* principal components correspond to the results obtained for $\boldsymbol{\mu}_1 = [1, 0.33, 2]$. The color scale is the same for the reference and the corresponding reconstructions.

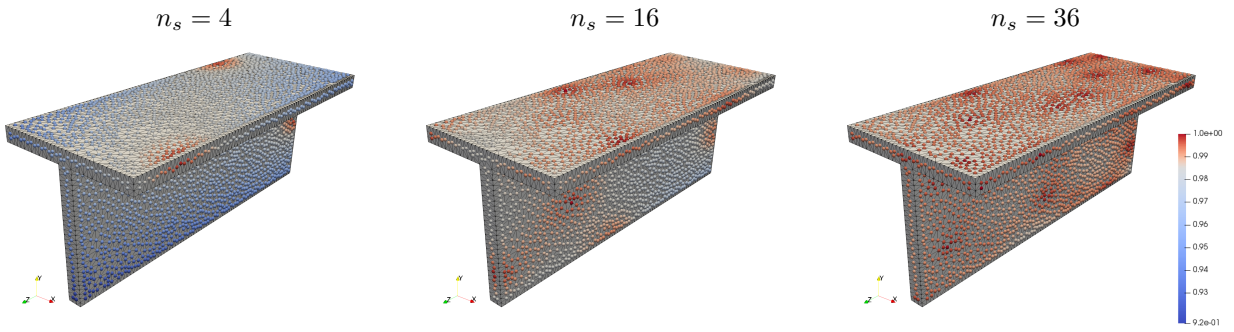


Figure 13: Relative variance reduction (15) obtained using n_s centroids and averaged over n_μ samples. Each plot shows a different fixed number n_s of inducing points, which increases from left to right, i.e. $n_s = 4, 16, 36$. The color scale is the same for the three plots.

In this way, the total number of degrees of freedom decreases to $N_h = 4632$. Here, we choose to first identify the optimal n_s components and then place one sensor per chosen component instead of computing the exact locations of the sensors directly. We note that this is a practical procedure in real-life engineering where the exact location of a sensor on a chosen component can be decided later, both empirically through engineering experience and practicality. We assume a budget of $n_s = 10$ displacement sensors and, for each one of the 192 components, we fix a sensor location, e.g., the point near the geometric center of that component. Thus, the admissible set Ω_s is such that $|\Omega_s| = 192$. We randomly generate $n_\mu = 40$ samples in Ω_μ and apply Algorithm 2 to get the n_s cluster centers as the components for sensor placement, as shown in Figure 15.

We note that though the geometry of the jacket structure is complicated, the chosen components are distributed approximately evenly over the whole domain, providing evidence that employing variational inference of sparse GPRs prevents waste of sensed information. To validate this sensor configuration, considering the complexity of the geometry and the large number of degrees of freedom, we return to the anomaly detection strategy introduced in Section 2.1. First, we place $n_s = 10$ displacement sensors on the surface of the optimal components and then train a one-class support vector machine (OC-SVM) classifier for each sensor location, following the procedure presented in [5], for $n_\mu = 100$ samples, randomly generated from Ω_μ. We observe that for real-life engineering problems, to assess the most probable damages, one may include know-how and experience of engineers. For the proposed configuration, we consider an increased wind load, i.e., $\Omega_\mu^{\text{extra}} = [1, 1.5]$ kPa, to represent a source of potential structural damages. We design four test scenarios, depending on the chosen input parameter space, i.e., either the baseline Ω_μ or the modified Ω_μ, and for each case we sample $n_\mu = 100$ parameters. In particular, case 1 corresponds to the healthy scenario, i.e., $\mu_x, \mu_y \in \Omega_\mu$; case 2 and 3 represent scenarios of potential minor damages, i.e., we choose $\mu_x \in \Omega_\mu^{\text{extra}}$ and $\mu_y \in \Omega_\mu$ for case 2 and, the opposite, i.e., $\mu_x \in \Omega_\mu$ and $\mu_y \in \Omega_\mu^{\text{extra}}$ for case 3; lastly, for case 4, the loads in both directions are sampled from the extended parameter space, i.e., $\mu_x, \mu_y \in \Omega_\mu^{\text{extra}}$. The trained OC-SVM classifiers are then used for testing, as shown in Table 1 and Table 2. We observe that under these conditions, the classifiers, trained with only $n_\mu = 100$ samples, provide accurate results for all four scenarios. However, among all scenarios, we observe that for case 2 we do not get as accurate results as compared to other cases. We point out that the test cases are randomly generated and we notice that the false positives in cases 2 and 3 correspond to the situation in which one of the two parameters, i.e., either μ_x or μ_y, sampled from $\Omega_\mu^{\text{extra}}$, is close to the lower bound, i.e., close to the healthy domain Ω_μ, fooling the classifier. In this case, the accuracy of the classifier can be improved by enlarging the training data set. Finally, we remark that, given the general situation where various types of anomalies in different locations can appear during the life time of a structure, relying on the assumption that we only have access to the simulation data of the healthy structure allows us to present a systematic way to place a designed amount of sensors to encourage the representation of the statistics of the whole domain while preventing sensed information waste.

6. Conclusions

A systematic approach to address the sensor placement problem in a SHM context where no prior knowledge on the damages is assumed is proposed. The examples presented in this work provide numerical evidence

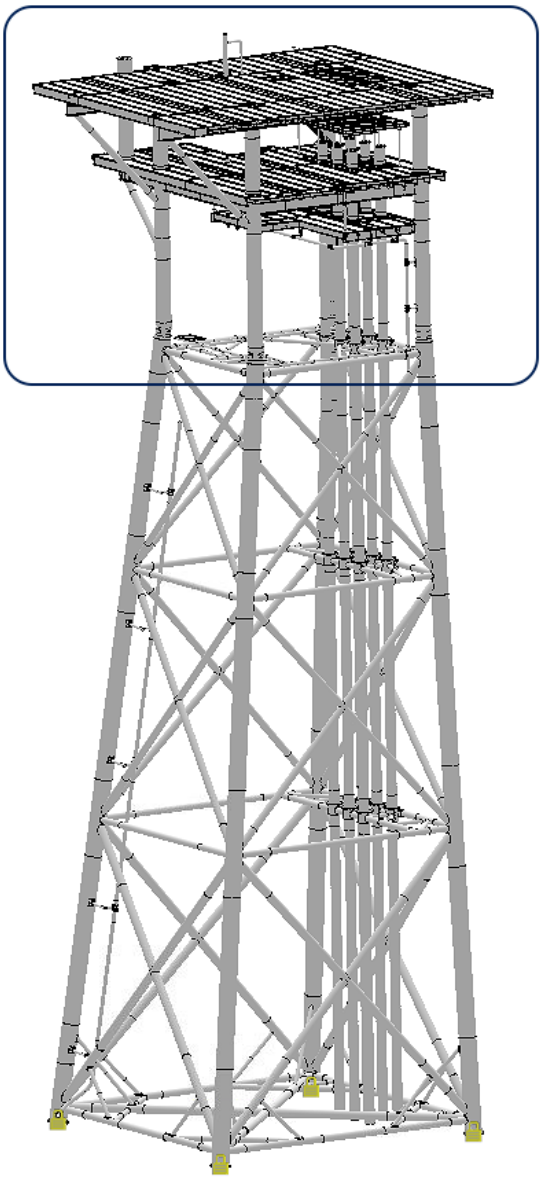


Figure 14: Jacket model: wind loads applied on components in the square.

that the variational inference of sparse GPR can be modified to place the sensors on structures characterized by complex geometries. The proposed approach is validated against both 2D and 3D numerical examples to confirm the quality of the sensor placement. We note that one of the novelties of the proposed method is that it does not assume any prior information of the anomalies, hence, it is robust to different type and severity of damages. In this work, the generation of synthetic healthy databases leverages reduced order modeling techniques to efficiently include physical and geometrical parametric dependencies. As a direct consequence, the method is easily extendable to other structures and avoids high computational costs related to simulating high fidelity models and considering all possible damage combinations.

We finally remark that in real-life engineering, the parameter space describing the natural variations of a large-scale structure is expected to be high dimensional. The procedure explained in this work can be extended to many parameters, but it requires a higher computational effort for both the construction of an healthy database and the training of multiple sparse variational GPR models. When the number of

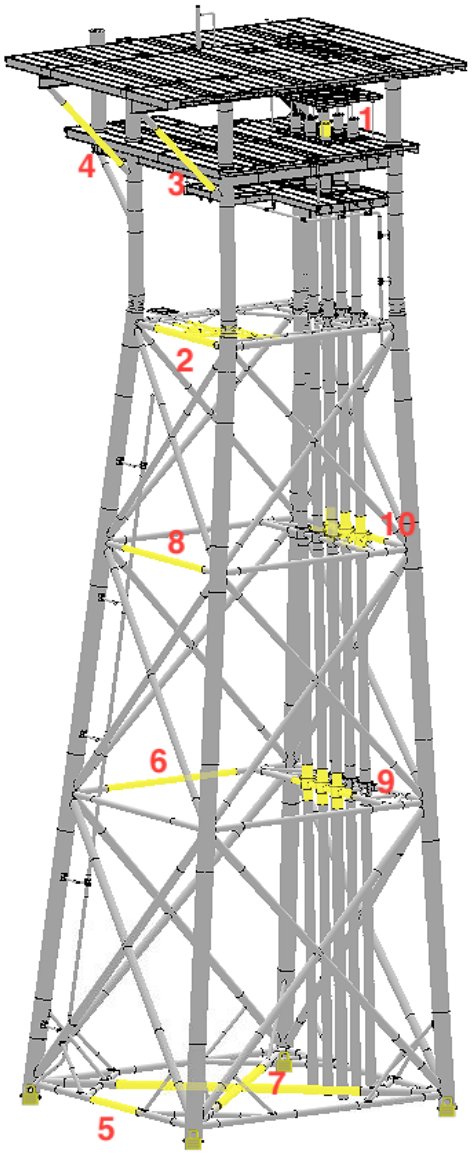


Figure 15: Jacket model: components chosen for sensor placement.

parameters is too large, one may rely on methodologies that compress the parameter space by retaining only those few parameters that influence the quantity of interest the most. The variance-based global sensitivity indices (Sobol's indices) [46] and the derivative based global sensitivity measures (DGSM) [26] are popular choices.

Acknowledgments. This work was partially supported by the Swiss Commission for Technology and Innovation (CTI) under Grant No. 25964.2 PFIW-IW.

References

[1] Akselos software. `https://akselos.com/`.

Sensor no.	Case 1	Case 2	Case 3	Case 4
1	99	81	93	100
2	100	83	94	100
3	99	77	94	100
4	100	77	94	100
5	98	91	92	100
6	100	81	93	100
7	100	79	90	100
8	99	74	94	100
9	100	83	93	100
10	100	76	93	100

Table 1: Sensor-wise percentages of accuracy for undamaged (case 1), minor damaged (cases 2 and 3) and major damaged (case 4) scenarios.

Sensor no.	Case 2	Case 3	Case 4
1	19	7	0
2	17	6	0
3	23	6	0
4	23	6	0
5	9	8	0
6	19	7	0
7	21	10	0
8	26	6	0
9	17	7	0
10	24	7	0

Table 2: Sensor-wise percentages of false positive samples for minor damaged (cases 2 and 3) and major damaged (case 4) scenarios.

[2] Argaud, J.-P., Bouriquet, B., De Caso, F., Gong, H., Maday, Y., and Mula, O. (2018). Sensor placement in nuclear reactors based on the Generalized Empirical Interpolation Method. *Journal of Computational Physics*, 363:354–370.

[3] Argyris, C., Chowdhury, S., Zabel, V., and Papadimitriou, C. (2018). Bayesian optimal sensor placement for crack identification in structures using strain measurements. *Structural Control and Health Monitoring*, 25(5):e2137.

[4] Arora, P., Varshney, S., et al. (2016). Analysis of K-means and K-medoids algorithm for big data. *Procedia Computer Science*, 78:507–512.

[5] Bigoni, C. and Hesthaven, J. S. (2020). Simulation-based anomaly detection and damage localization: an application to Structural Health Monitoring. *Computer Methods in Applied Mechanics and Engineering*, 363:112896.

[6] Brunton, B. W., Brunton, S. L., Proctor, J. L., and Kutz, J. N. (2016). Sparse sensor placement optimization for classification. *SIAM Journal on Applied Mathematics*, 76(5):2099–2122.

[7] Capellari, G., Chatzi, E., Mariani, S., and Azam, S. E. (2017). Optimal design of sensor networks for damage detection. *Procedia engineering*, 199:1864–1869.

[8] Chandola, V., Banerjee, A., and Kumar, V. (2009). Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):1–58.

[9] Chen, Z. and Wang, B. (2018). How priors of initial hyperparameters affect Gaussian process regression models. *Neurocomputing*, 275:1702–1710.

[10] Cressie, N. (1991). *Statistics for spatial data*. A Wiley-interscience publication. J. Wiley.

[11] Csató, L. and Opper, M. (2002). Sparse on-line Gaussian processes. *Neural computation*, 14(3):641–668.

[12] Davis, L. (1991). Handbook of genetic algorithms.

[13] Eftang, J. L. and Patera, A. T. (2014). A port-reduced static condensation reduced basis element method for large component-synthesized structures: approximation and a posteriori error estimation. *Advanced Modeling and Simulation in Engineering Sciences*, 1(1):3.

[14] Farrar, C. R. and Worden, K. (2012). *Structural Health Monitoring: a machine learning perspective*. John Wiley & Sons.

[15] Flynn, E. B. and Todd, M. D. (2010). A Bayesian approach to optimal sensor placement for Structural Health Monitoring with application to active sensing. *Mechanical Systems and Signal Processing*, 24(4):891–903.

[16] Fortin, F.-A., Rainville, F.-M. D., Gardner, M.-A., Parizeau, M., and Gagné, C. (2012). DEAP: Evolutionary algorithms made easy. *Journal of Machine Learning Research*, 13(Jul):2171–2175.

[17] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep learning*.

[18] GPy (since 2012). GPy: A Gaussian process framework in python.

[19] Graves, A., Jaitly, N., and Mohamed, A.-r. (2013). Hybrid speech recognition with deep bidirectional lstm. In *2013 IEEE workshop on automatic speech recognition and understanding*, pages 273–278. IEEE.

[20] Guo, H., Zhang, L., Zhang, L., and Zhou, J. (2004). Optimal placement of sensors for Structural Health Monitoring using improved genetic algorithms. *Smart materials and structures*, 13(3):528.

[21] Hesthaven, J. S., Rozza, G., and Stamm, B. (2015). *Certified Reduced Basis Methods for Parametrized Partial Differential Equations*. SpringerBriefs in Mathematics. Springer International Publishing.

[22] Huynh, D. B. P., Knezevic, D. J., and Patera, A. T. (2013). A static condensation reduced basis element method: approximation and a posteriori error estimation. *ESAIM: Mathematical Modelling and Numerical Analysis*, 47(1):213–251.

[23] Joe, S. and Kuo, F. Y. (2008). Constructing Sobol sequences with better two-dimensional projections. *SIAM Journal on Scientific Computing*, 30(5):2635–2654.

[24] Kapteyn, M. G., Knezevic, D. J., and Willcox, K. (2020). Toward predictive digital twins via component-based reduced-order models and interpretable machine learning. In *AIAA Scitech 2020 Forum*, page 0418.

[25] Krause, A., Singh, A., and Guestrin, C. (2008). Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research*, 9(Feb):235–284.

[26] Kucherenko, S. et al. (2009). Derivative based global sensitivity measures and their link with global sensitivity indices. *Mathematics and Computers in Simulation*, 79(10):3009–3017.

[27] Lecerf, M., Allaire, D., and Willcox, K. (2015). Methodology for dynamic data-driven online flight capability estimation. *AIAA Journal*, 53(10):3073–3087.

[28] Lee, B. and Staszewski, W. (2007). Sensor location studies for damage detection with Lamb waves. *Smart materials and structures*, 16(2):399.

[29] Long, J. and Buyukozturk, O. (2014). Automated Structural Damage Detection Using One-Class Machine Learning. In *Dynamics of Civil Structures, Volume 4*, pages 117–128. Springer.

[30] MATLAB (2018). *Version 9.5 (R2018b)*. The MathWorks Inc., Natick, Massachusetts.

[31] Michaels, J. E. and Michaels, T. E. (2007). Guided wave signal processing and image fusion for in situ damage localization in plates. *Wave motion*, 44(6):482–492.

[32] Murphy, K. P. (2012). *Machine learning: a probabilistic perspective*. MIT press.

[33] Ostachowicz, W., Soman, R., and Malinowski, P. (2019). Optimization of sensor placement for Structural Health Monitoring: A review. *Structural Health Monitoring*, page 1475921719825601.

[34] Papadimitriou, C. (2004). Optimal sensor placement methodology for parametric identification of structural systems. *Journal of sound and vibration*, 278(4-5):923–947.

[35] Park, H.-S. and Jun, C.-H. (2009). A simple and fast algorithm for K-medoids clustering. *Expert systems with applications*, 36(2):3336–3341.

[36] Pimentel, M. A., Clifton, D. A., Clifton, L., and Tarassenko, L. (2014). A review of novelty detection. *Signal Processing*, 99:215–249.

[37] Quaranta, G., Lopez, E., Abisset-Chavanne, E., Duval, J. L., Huerta, A., and Chinesta, F. (2019). Structural health monitoring by combining machine learning and dimensionality reduction techniques. *Revista internacional de métodos numéricos para cálculo y diseño en ingeniería*, 35(1).

[38] Quarteroni, A., Manzoni, A., and Negri, F. (2015). *Reduced Basis Methods for Partial Differential Equations: An Introduction*, volume 92. Springer.

[39] Quiñonero-Candela, J. and Rasmussen, C. E. (2005). A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research*, 6(Dec):1939–1959.

[40] Rosafalco, L., Manzoni, A., Mariani, S., and Corigliano, A. (2020). Fully convolutional networks for structural health monitoring through multivariate time series classification. *arXiv preprint arXiv:2002.07032*.

[41] Seeger, M., Williams, C., and Lawrence, N. (2003). Fast forward selection to speed up sparse Gaussian process regression. Technical report.

[42] Sivanandam, S. and Deepa, S. (2008). Genetic algorithms. In *Introduction to genetic algorithms*, pages 15–37. Springer.

[43] Smola, A. J. and Bartlett, P. L. (2001). Sparse greedy Gaussian process regression. In *Advances in neural information processing systems*, pages 619–625.

[44] Smola, A. J. and Schölkopf, B. (2000). Sparse greedy matrix approximation for machine learning.

[45] Snelson, E. and Ghahramani, Z. (2006). Sparse Gaussian processes using pseudo-inputs. In *Advances in neural information processing systems*, pages 1257–1264.

[46] Sobol, I. M. (2001). Global sensitivity indices for nonlinear mathematical models and their monte carlo estimates. *Mathematics and computers in simulation*, 55(1-3):271–280.

[47] Soman, R., Malinowski, P., Kudela, P., and Ostachowicz, W. (2018). Analytical, numerical and experimental formulation of the sensor placement optimization problem for guided waves. In *Proceedings of the 9th EWSHM Conference, Manchester, UK*, pages 10–13.

[48] Taddei, T., Penn, J., Yano, M., and Patera, A. (2018). Simulation-based classification; a model-order-reduction approach for Structural Health Monitoring. *Archives of Computational Methods in Engineering*, 25(1):23–45.

[49] Thiene, M., Khodaei, Z. S., and Aliabadi, M. (2016). Optimal sensor placement for maximum area coverage (MAC) for damage localization in composite structures. *Smart materials and structures*, 25(9):095037.

[50] Titsias, M. (2009). Variational learning of inducing variables in sparse Gaussian processes. In *Artificial Intelligence and Statistics*, pages 567–574.

[51] Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., et al. (2020). Scipy 1.0: fundamental algorithms for scientific computing in python. *Nature methods*, pages 1–12.

[52] Williams, C. K. and Rasmussen, C. E. (1996). Gaussian processes for regression. In *Advances in neural information processing systems*, pages 514–520.

[53] Williams, C. K. and Rasmussen, C. E. (2006). *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA.