Predictive Monitoring of Large-Scale Engineering Assets Using Machine Learning Techniques and Reduced-Order Modeling

Caterina Bigoni, Mengwu Guo, and Jan S. Hesthaven

Abstract Structural health monitoring techniques aim at providing an automated solution to the threat of unsurveilled aging of structures that can have tremendous consequences in terms of fatalities, environmental pollution, and economic loss. To assess the state of damage of a complex structure, this paper proposes to fully characterize its behavior under multiple environmental and operational scenarios and compare new sensor measurements with the baseline behavior. However, the repeated simulations of a nonlinear, time-dependent structural model with high-dimensional input parameters represent a severe computational bottleneck for large-scale engineering assets. This chapter presents how to use efficient reduced-order modeling techniques to mitigate the computational effort of many-query simulations without jeopardizing the accuracy. To compare new sensor measurements with the natural behavior of synthetic solutions, the proposed methodology uses hierarchical semi-supervised learning algorithms on a small amount of extracted damage-sensitive features, thus allowing one to assess the state of damage in real time. Using the inexpensive simulations, one can also optimally place sensors to maximize the observability of discriminant features. The all-round methodology is validated on a numerical example.

Keywords Simulation-based anomaly detection · Reduced order models · One-class classification · Sensor placement · Structural health monitoring · Digital twins

C. Bigoni (✉) · J. S. Hesthaven
Computational Mathematics and Simulation Science (MCSS), Ecole Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland
e-mail: caterina.bigoni@alumni.epfl.ch

J. S. Hesthaven
e-mail: Jan.Hesthaven@epfl.ch

M. Guo
Department of Applied Mathematics, University of Twente, Enschede, Netherlands
e-mail: m.guo@utwente.nl

A. Cury et al. (eds.), *Structural Health Monitoring Based on Data Science Techniques*, Structural Integrity 21, https://doi.org/10.1007/978-3-030-81716-9_9

1 Introduction

Many existing private and public assets, such as civil engineering infrastructures, buildings, or aircraft, require reliable damage detection techniques to be safely used, especially during their inevitable aging. When monitoring a structure over its lifecycle, its deterioration and damages represent a great concern and the early detection of critical decay might prevent failures that can cause sudden shutdowns or even catastrophes with severe life-safety and economic repercussions [1]. To prevent these critical failures from happening, techniques of structural health monitoring (SHM) have been developed in recent studies with applications to civil and aerospace engineering, as well as to the conservation of cultural heritage structures. SHM refers to automated monitoring procedures that seek to provide reliable information on the performance and integrity of a structure in real time. In the context of SHM, the combination of sensor measurements, numerical models simulating the underlying behavior of a structure of interest under different environmental and operational conditions, and machine learning techniques has led to the design of structural *digital twins* [2].

The focus of this work is on *wave propagation* approaches to data-driven, predictive SHM, which aim to detect damages by examining the distortions in propagating elastic waves as a result of reflections and amplitude attenuations when intersecting the damage boundary. Featuring a data-driven nature, these approaches, sometimes called *simulation-based* SHM [3–7], are decomposed into an offline phase and an online phase. In the former, a database of synthetic signals is built to represent the structural behavior under different conditions, while in the latter real experimental time signals, collected from sensors placed on a structure, are compared with those simulated offline using a classifier that discriminates between damaged and undamaged states.

Toward an efficient and robust scheme of data-driven predictive monitoring, reduced-order modeling techniques are integrated into a wave propagation approach with cutting-edge machine learning tools. The data-driven SHM setting corresponds to a multiquery problem, where one has to solve high-dimensional, time-dependent, parametric equations, which results in great demands for computational resources. To overcome such a computational burden, model order reduction is employed to project the original full-order system onto a reduced space with a significantly lower dimensionality. In this way, a robust dataset of approximated sensor measurements is generated. In SHM, the task of damage detection is typically reduced to a supervised learning process relying on a fully labeled dataset, obtained from both healthy and damaged structures (either generated with computer-aided procedures or collected experimentally). However, gathering an exhaustive collection of configuration classes anticipating all types of damages is typically unrealistic and the number of different classification labels may grow rapidly. Instead, this work relies on semi-supervised learning techniques, also called *one-class classification* methods, which learn the common features among labeled data belonging to the *normal* class in the training phase. Unlabeled data from both classes are then used in the test phase to identify abnormal data which deviate from the normal mode. One-class algorithms allow to locate the damage and estimate its severity by training a different model for each sensor location. Finally, guided by an appropriate indicator of the damage detection performance, a modified sparse Gaussian process method is applied to the synthetic dataset of healthy configurations to systematically place a fixed number of sensors on a structure of interest.

Following the introduction, a model problem for sensor measurements governed by the acoustic-elastic equation is briefly reviewed in Sect. 2. Basic techniques of reduced-order modeling, which can be used for the multiquery simulations in SHM, are introduced in Sect. 3. A local semi-supervised method is used for automatic anomaly detection in Sect. 4, and the variational sparse Gaussian process model is utilized for optimal sensor placement in Sect. 5. In Sect. 6, the all-round methodology is demonstrated by a numerical example. Finally, conclusions are drawn in Sect. 7.

2 A Model for Sensor Measurements

This section introduces the model problem for sensor measurements, including its governing equations and parametric discrete formulations.

2.1 Governing Equation

Let $\Omega \subset \mathbb{R}^{d_\Omega}$ be a polygonal physical domain with piece-wise smooth boundary $\partial\Omega$, where $d_\Omega = 2, 3$ is the spatial dimension, and let $[0, T]$, with $T \in \mathbb{R}_+$, be a suitable time interval. Here, Ω represents a structure of interest and $[0, T]$ a suitable time window to observe the response of a structure undergoing a predefined excitation (i.e., the effect of an active or passive source) through sensor measurements. Moreover, let $\mathcal{P} \subset \mathbb{R}^{d_\mu}$ be a suitable parameter domain, where d_μ indicates the number of input parameters required to describe the healthy variations that a structure may undergo during its life time, and let $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_{d_\mu})$ be a parameter vector representing one possible healthy variation of the environmental and operational conditions, i.e., μ_i may relate to the material properties, the boundary conditions, the initial conditions, or the source function for $1 \leq i \leq d_\mu$.

Let $\boldsymbol{u} = \boldsymbol{u}(\boldsymbol{x}, t; \boldsymbol{\mu}) : \Omega \times [0, T] \times \mathcal{P} \to \mathbb{R}^{d_\Omega}$ be a vector-valued displacement field, solution of the acoustic-elastic equation equipped with suitable boundary and initial conditions:

$$\begin{cases} \rho(\ddot{\boldsymbol{u}} + \eta\dot{\boldsymbol{u}}) - \nabla \cdot \boldsymbol{\sigma} = h(t; \boldsymbol{\mu})\boldsymbol{b}(\boldsymbol{x}; \boldsymbol{\mu}) & \text{in } \Omega \\ \boldsymbol{u} \cdot \boldsymbol{n} = \boldsymbol{0}, \quad (\boldsymbol{\sigma} \cdot \boldsymbol{n}) \cdot \boldsymbol{\tau} = \boldsymbol{t}_N & \text{on } \partial\Omega \\ \boldsymbol{u}|_{t=0} = \boldsymbol{u}_0, \quad \dot{\boldsymbol{u}}|_{t=0} = \boldsymbol{v}_0 & \text{in } \Omega \end{cases}, \tag{1}$$

where $\ddot{\boldsymbol{u}} = \partial^2 \boldsymbol{u}/\partial t^2$ and $\dot{\boldsymbol{u}} = \partial \boldsymbol{u}/\partial t$ are the acceleration and velocity fields. Here, ρ is the density coefficient, η is a dimensionless damping coefficient, $h : [0, T] \times \mathcal{P} \to \mathbb{R}$ and $\boldsymbol{b} : \Omega \times \mathcal{P} \to \mathbb{R}^{d_\Omega}$ are two source functions depending on time and space, respectively, $\boldsymbol{\sigma} = \boldsymbol{\sigma}(\boldsymbol{u}; \boldsymbol{\mu})$ is the stress tensor, $\boldsymbol{n}$ and $\boldsymbol{\tau}$ are the outward normal and tangential (unit) vectors to $\partial\Omega$, respectively, $\boldsymbol{t}_N = \boldsymbol{t}_N(\boldsymbol{x}, t; \boldsymbol{\mu})$ is the traction vector used in the definition of the free-slip boundary conditions, $\boldsymbol{u}_0 = \boldsymbol{u}_0(\boldsymbol{x}; \boldsymbol{\mu})$ and $\boldsymbol{v}_0 = \boldsymbol{v}_0(\boldsymbol{x}; \boldsymbol{\mu})$ describe the initial displacement and velocity in space, respectively.

The ultimate goal is to emulate the real sensor response at m given sensor locations $\boldsymbol{x}_i \in \Omega$ for $1 \leq i \leq m$. To do so, let $\ell : \mathbb{R}^{d_\Omega} \times \mathcal{P} \to \mathbb{R}^{d_g}$ be an input–output function and $g_i : [0, T] \times \mathcal{P} \to \mathbb{R}^{d_g}$ a (parametric) output of interest, i.e., an approximation of the sensor response at time t and location $\boldsymbol{x}_i$:

$$g_i(t; \boldsymbol{\mu}) = \ell(\boldsymbol{u}(\boldsymbol{x}_i, t; \boldsymbol{\mu}); \boldsymbol{\mu}), \quad 1 \leq i \leq m. \tag{2}$$

Before proceeding with classic discretization techniques such as the finite element method, consider the vector space $\mathbb{V} = \{\boldsymbol{w} \in [H^1(\Omega)]^{d_\Omega} : \boldsymbol{w} \cdot \boldsymbol{n} = \boldsymbol{0} \text{ on } \partial\Omega\}$, equipped with a suitable inner product $\langle \cdot, \cdot \rangle_{\mathbb{V}}$ and the corresponding induced norm $\|\cdot\|_{\mathbb{V}}$. Moreover, consider a parametrized linear form $f : \mathbb{V} \times \mathcal{P} \to \mathbb{R}$ where the linearity is with respect to the first argument, and the parametrized bilinear forms $m : \mathbb{V} \times \mathbb{V} \to \mathbb{R}$ and $a : \mathbb{V} \times \mathbb{V} \times \mathcal{P} \to \mathbb{R}$, where the bilinearity is with respect to the first two arguments. Then, the acoustic-elastic problem in abstract form reads: given $t \in [0, T]$ and $\boldsymbol{\mu} \in \mathcal{P}$, find $\boldsymbol{u} = \boldsymbol{u}(t; \boldsymbol{\mu}) \in \mathbb{V}$ such that

$$\begin{cases} \rho \left(m(\ddot{\boldsymbol{u}}, \boldsymbol{\psi}) + \eta\, m(\dot{\boldsymbol{u}}, \boldsymbol{\psi}) \right) + a(\boldsymbol{u}, \boldsymbol{\psi}; \boldsymbol{\mu}) = h(t; \boldsymbol{\mu}) f(\boldsymbol{\psi}; \boldsymbol{\mu}) \quad \forall \boldsymbol{\psi} \in \mathbb{V}, \\ \boldsymbol{u}(t = 0; \boldsymbol{\mu}) = \boldsymbol{0}, \quad \dot{\boldsymbol{u}}(t = 0; \boldsymbol{\mu}) = \boldsymbol{0}, \end{cases} \tag{3}$$

with

$$\begin{aligned} m(\boldsymbol{u}, \boldsymbol{\psi}) &= \int_\Omega \boldsymbol{u} \cdot \boldsymbol{\psi}\, \boldsymbol{\Omega}, \quad f(\boldsymbol{\psi}; \boldsymbol{\mu}) = \int_\Omega \boldsymbol{b}(\boldsymbol{\mu}) \cdot \boldsymbol{\psi}\, \boldsymbol{\Omega}, \\ a(\boldsymbol{u}, \boldsymbol{\psi}; \boldsymbol{\mu}) &= \int_\Omega [\boldsymbol{H}(\boldsymbol{\mu}) : \boldsymbol{\varepsilon}(\boldsymbol{u})] : \boldsymbol{\varepsilon}(\boldsymbol{\psi})\, \boldsymbol{\Omega}, \end{aligned} \tag{4}$$

in which $\boldsymbol{H}$ is the Hooke's stiffness tensor and $\boldsymbol{\varepsilon}(\cdot) = \frac{1}{2}[\nabla \cdot + (\nabla \cdot)^{\mathrm{T}}]$ is the operator of Cauchy strain. Note that, for the sake of simplicity, homogeneous boundary and initial conditions in (3) are considered, i.e., $\boldsymbol{u}_0 = \boldsymbol{0}$, $\boldsymbol{v}_0 = \boldsymbol{0}$, and $\boldsymbol{t}_N = \boldsymbol{0}$.

2.2 Parametric Discrete Problem

This section introduces a discrete approximation space $\mathbb{V}_h \subset \mathbb{V}$ as well as a discretization of the time interval $[0, T]$ in which the approximate solution is sought. The approximation space $\mathbb{V}_h$ is here constructed by a standard finite element method based on piece-wise linear basis functions and a triangulation of Ω, i.e., non-overlapping triangles ($d_\Omega = 2$) or tetrahedra ($d_\Omega = 3$) whose union perfectly coincides with Ω. Alternative discretization strategies include spectral methods or higher-order finite elements. Let $\mathbb{V}_h$ be equipped with a basis $\{\boldsymbol{\varphi}_j(\boldsymbol{x}) \in \mathbb{R}^{d_\Omega}\}_{j=1}^{N_h}$, where $N_h = \dim(\mathbb{V}_h)$ is the number of degrees of freedom (DOFs). Moreover, divide the time interval $[0, T]$ into N_t subintervals of equal length $\Delta t = \frac{T}{N_t}$ and define $t^n = n\Delta t,\ 0 \le n \le N_t$.

The discrete problem with finite element discretization seeks to find $\boldsymbol{u}_h(t; \boldsymbol{\mu}) \in \mathbb{V}_h$, which can be expressed as $\boldsymbol{u}_h(t; \boldsymbol{\mu}) = \sum_{j=1}^{N_h} (\boldsymbol{u}_h(t; \boldsymbol{\mu}))_j \boldsymbol{\varphi}_j(\boldsymbol{x})$ where $(\boldsymbol{u}_h)_j$ denotes the jth entry of the solution vector $\boldsymbol{u}_h \in \mathbb{R}^{N_h}$. With an additional discretization over time, one can retrieve the solution vector at the nth time step, denoted by $\boldsymbol{u}_h^n(\boldsymbol{\mu}) = \boldsymbol{u}_h(t^n; \boldsymbol{\mu}),\ n = 1, \dots, N_t$. Moreover, let $\boldsymbol{v}_h^n(\boldsymbol{\mu}) \in \mathbb{R}^{N_h}$ and $\boldsymbol{a}_h^n(\boldsymbol{\mu}) \in \mathbb{R}^{N_h}$ be the velocity and the acceleration vectors, respectively, such that their elements are the multiplicative coefficients of the following expressions: $\dot{\boldsymbol{u}}_h^n(\boldsymbol{\mu}) = \sum_{j=1}^{N_h} (\boldsymbol{v}_h^n(\boldsymbol{\mu}))_j \boldsymbol{\varphi}_j(\boldsymbol{x})$ and $\ddot{\boldsymbol{u}}_h^n(\boldsymbol{\mu}) = \sum_{j=1}^{N_h} (\boldsymbol{a}_h^n(\boldsymbol{\mu}))_j \boldsymbol{\varphi}_j(\boldsymbol{x})$, respectively.

Once the acoustic-elastic equation is spatially discretized by finite elements, the corresponding algebraic formulation is written as follows for a given $\boldsymbol{\mu} \in \mathcal{P}$ and $t \in [0, T]$:

$$\rho \boldsymbol{M}_h \left[\ddot{\boldsymbol{u}}_h(\boldsymbol{\mu}) + \eta \dot{\boldsymbol{u}}_h(\boldsymbol{\mu})\right] + \boldsymbol{A}_h(\boldsymbol{\mu}) \boldsymbol{u}_h(\boldsymbol{\mu}) = h(t; \boldsymbol{\mu}) \boldsymbol{f}_h(\boldsymbol{\mu}), \tag{5}$$

where $\boldsymbol{M}_h \in \mathbb{R}^{N_h \times N_h}$ is the mass matrix, $\boldsymbol{A}_h(\boldsymbol{\mu}) \in \mathbb{R}^{N_h \times N_h}$ the parametrized stiffness matrix, and $\boldsymbol{f}_h(\boldsymbol{\mu}) \in \mathbb{R}^{N_h}$ the parametrized right-hand side vector with entries

$$\begin{aligned} (\boldsymbol{M}_h)_{ij} &= m(\boldsymbol{\varphi}_j, \boldsymbol{\varphi}_i), \quad (\boldsymbol{A}_h(\boldsymbol{\mu}))_{ij} = a(\boldsymbol{\varphi}_j, \boldsymbol{\varphi}_i; \boldsymbol{\mu}), \\ \text{and} \quad (\boldsymbol{f}_h(\boldsymbol{\mu}))_i &= f(\boldsymbol{\varphi}_i; \boldsymbol{\mu}), \quad 1 \le i, j \le N_h. \end{aligned} \tag{6}$$

A classic Newmark method is then applied to the temporal discretization and the governing equation becomes: given $\boldsymbol{\mu} \in \mathcal{P}$, find the acceleration vector $\boldsymbol{a}_h^n(\boldsymbol{\mu}) \in \mathbb{R}^{N_h}$ for $n = 1, \dots, N_t$ such that

$$\left[\rho(1 + \eta\zeta\Delta t)\boldsymbol{M}_h + \beta(\Delta t)^2 \boldsymbol{A}_h(\boldsymbol{\mu})\right] \boldsymbol{a}_h^n(\boldsymbol{\mu}) = h(t^n; \boldsymbol{\mu}) \boldsymbol{f}_h(\boldsymbol{\mu}) - \boldsymbol{q}_h^{n-1}(\boldsymbol{\mu}), \tag{7}$$

in which β and ζ are two constant parameters, here chosen as $\zeta = 2\beta = 2$, which corresponds to a popular second-order method [8, 9], while $\boldsymbol{q}_h^{n-1}(\boldsymbol{\mu}) \in \mathbb{R}^{N_h}$ is given as

$$\begin{aligned} \boldsymbol{q}_h^{n-1}(\boldsymbol{\mu}) =& \boldsymbol{A}_h(\boldsymbol{\mu}) \boldsymbol{u}_h^{n-1}(\boldsymbol{\mu}) + (\rho\eta \boldsymbol{M}_h + \Delta t \boldsymbol{A}_h(\boldsymbol{\mu})) \boldsymbol{v}_h^{n-1}(\boldsymbol{\mu}) \\ &+ \left(\rho\eta(1 - \zeta)\Delta t \boldsymbol{M}_h + \tfrac{1-2\beta}{2}(\Delta t)^2 \boldsymbol{A}_h(\boldsymbol{\mu})\right) \boldsymbol{a}_h^{n-1}(\boldsymbol{\mu}). \end{aligned} \tag{8}$$

Finally, the displacement solution vector $\boldsymbol{u}_h^n(\boldsymbol{\mu})$ is obtained using the updating rule of the implicit Newmark method, introduced in [10] and defined as:

$$\boldsymbol{u}_h^n(\boldsymbol{\mu}) = \boldsymbol{u}_h^{n-1}(\boldsymbol{\mu}) + \Delta t \boldsymbol{v}_h^{n-1}(\boldsymbol{\mu}) \left(\beta \boldsymbol{a}_h^n(\boldsymbol{\mu}) + \tfrac{1-2\beta}{2} \boldsymbol{a}_h^{n-1}(\boldsymbol{\mu})\right) \tag{9}$$

$$\boldsymbol{v}_h^n(\boldsymbol{\mu}) = \boldsymbol{v}_h^{n-1}(\boldsymbol{\mu}) + \Delta t \left(\zeta \boldsymbol{a}_h^n(\boldsymbol{\mu}) + (1-\zeta) \boldsymbol{a}_h^{n-1}(\boldsymbol{\mu})\right), \tag{10}$$

Problem (7) is denoted as the *truth problem* and $\boldsymbol{u}_h^n(\boldsymbol{\mu})$ as the *truth solution* at n-th time step, which, in principle, can be achieved with as high accuracy as desired. However, many degrees of freedom may be involved in the problem, thus leading to a computationally expensive method due to inversion of the N_h-dimensional matrix in the left-hand side of (7). In addition, to fully represent the healthy variations of the structure, one needs to estimate an approximation to the output of interest (2) for many input parameter values $\{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \dots\}$ over the whole discrete time window $0 = t^0, t^1, \dots, t^{N_t} = T$, i.e.,

$$\boldsymbol{g}_i^k = \left[g_i(t^0; \boldsymbol{\mu}_k), g_i(t^1; \boldsymbol{\mu}_k), \dots, g_i(t^{N_t}; \boldsymbol{\mu}_k)\right], \quad k = 1, 2, \dots, \tag{11}$$

evaluated at all the sensor locations $\boldsymbol{x}_i$, $1 \leq i \leq m$. For each input parameter value $\boldsymbol{\mu}_k$, the total computational cost involves the resolution of N_t linear systems of dimension N_h.

3 Techniques of Reduced-Order Modeling

When the dimensionality of a full-order system, defined as N_h in Sect. 2, is large, the repeated solution of such a time-dependent problem with varying input parameters can result in great demands on both CPU time and memory, which is often computationally prohibitive. To reduce the computational cost without significantly compromising the overall accuracy, reduced-order models (ROMs) have been developed. In general, reduced-order modeling seeks to find a low-dimensional representation of the full-order solution manifold and hence reduce the dimensionality by projecting the original governing equations onto a low-dimensional space.

The reduced basis (RB) method [11, 12] is a typical projection-based approach to ROMs and features an offline–online framework. With a significantly smaller dimension than the full-order model, a reduced space is spanned by a set of RB modes that are extracted offline from a collection of full-order snapshots at several time-parameter locations. Once the RB space is constructed, the approximate solution for an unseen parameter value is recovered online in the reduced space. Conventionally, a Galerkin projection is adopted to determine the combination coefficients associated with the RB, yielding the reduced-order solutions during the online stage.

For time-dependent problems, due to the traveling-wave behavior of the solution, classic projection-based ROM strategies [13] may pose several challenges, e.g., the manifold of all possible solutions can often not be compressed to a small reduced basis. Furthermore, the sampling strategy is more complicated since it has to combine the solution at different time instants and for different values of the parameter. For example, the readers can refer to the POD-greedy sampling strategy [14] and the randomized SVD algorithms [15]. Recent efforts have been made in the direction of space–time approaches, where projection in space and time is performed simultaneously, see, e.g., [16]. A different effective strategy is to replace the time domain formulation with a frequency domain formulation and to apply a ROM method to replace the full-order problem in frequency domain. In this way, the number of time instances N_t where one expects to solve a linear system equivalent to (7) is reduced to a number of principal frequencies N_z, with $N_z \ll N_t$. In addition, recurring to a ROM strategy reduces the number of degrees of freedom of each linear system to the size of reduced basis, i.e., from N_h to r, with $r \ll N_h$. Without going in too much detail, the reader are referred to [5, 17], where the authors, motivated by the interest of studying the transient response of damaged structures under the effect active sources, construct a reduced-order model of the acoustic-elastic equation in the Laplace domain.

The goal here is to provide a brief introduction to several basic elements of the RB method, which lay the foundation for more advanced techniques. In particular, after introducing a general formulation of the proper orthogonal decomposition (POD) in Sect. 3.1, the construction of RB from full-order snapshots using the POD is also described. The technique to retrieve the RB solution, an approximation of the high-fidelity solution, is ultimately presented in Sect. 3.2.

3.1 Proper Orthogonal Decomposition

General formulation of the POD

In a vector space $\mathbb{X}$, equipped with an inner product $\langle \cdot, \cdot \rangle_{\mathbb{X}}$, consider a collection of snapshot vectors, denoted by $\{p_1, \ldots, p_{N_s}\} \subset \mathbb{X}$. A correlation matrix $\boldsymbol{C} \in \mathbb{R}^{N_s \times N_s}$ of the snapshots is formed as

$$\boldsymbol{C}_{ij} = \langle p_i, p_j \rangle_{\mathbb{X}}, \quad 1 \leq i, j \leq N_s. \tag{12}$$

The eigenvalue problem of such a correlation matrix $\boldsymbol{C}$ is then written as

$$\boldsymbol{C}\boldsymbol{z}^{(i)} = \lambda^{(i)} \boldsymbol{z}^{(i)}, \quad 1 \leq i \leq N_s, \tag{13}$$

in which $\lambda^{(1)} \geq \cdots \geq \lambda^{(N_s)} \geq 0$. By taking

$$\phi_i = \sum_{j=1}^{N_s} p_j \left(\boldsymbol{z}^{(i)}\right)_j / \sqrt{\lambda^{(i)}}, \quad 1 \leq i \leq r \text{with} \, N \leq N_s, \tag{14}$$

an orthonormal basis is formed, i.e., $\langle \phi_i, \phi_j \rangle = \delta_{ij}, \ 1 \leq i, j \leq r$, and an r-dimensional subspace is then constructed as $\mathbb{X}_r = \text{span}\{\phi_1, \ldots, \phi_N\} \subset \mathbb{X}$. The projection onto this subspace, denoted by $P_N[\cdot] : \mathbb{X} \to \mathbb{X}_r$, is thus defined as

$$P_r[f] = \arg\min_{\xi \in \mathbb{X}_r} \|f - \xi\|_{\mathbb{X}}^2 = \sum_{i=1}^{r} \langle f, \phi_i \rangle_{\mathbb{X}} \phi_i, \quad f \in \mathbb{X}. \tag{15}$$

It can be shown that the projection error of the snapshots only depends on the truncated eigenvalues, written as

$$\sum_{i=1}^{N_s} \|p_i - P_r[p_i]\|_{\mathbb{X}}^2 = \sum_{i=r+1}^{N_s} \lambda^{(i)}. \tag{16}$$

In addition, $\mathbb{X}_r$ is the optimal subspace of $\mathbb{S} = \text{span}\{p_1, \ldots, p_{N_s}\}$ that minimizes the projection error, i.e.,

$$\sum_{i=1}^{N_s} \|p_i - P_r[p_i]\|_{\mathbb{X}}^2 = \min_{\mathbb{U} \text{ being a subspace of } \mathbb{S}} \left\{ \sum_{i=1}^{N_s} \min_{\xi \in \mathbb{U}} \|p_i - \xi\|_{\mathbb{X}}^2 \right\}. \tag{17}$$

Construction of RB using the POD

At the algebraic level, the solution space for the full-order discrete system (5) is $\mathbb{R}^{N_h}$, i.e., $\mathbb{X} = \mathbb{R}^{N_h}$, and it is correspondingly equipped with the Euclidean inner product. To construct an RB space, one has to collect the solution snapshots of N_t time instances $\{t^0, t^1, \ldots, t^{N_t}\}$ at N_μ parameter locations $\{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_{N_\mu}\}$, i.e., $\{p_i\}_{i=1}^{N_s} = \{\boldsymbol{u}_h^n(\boldsymbol{\mu}_k) : 1 \le n \le N_t, 1 \le k \le N_\mu\}$ and $N_s = N_t N_\mu$. Let $\boldsymbol{S} \in \mathbb{R}^{N_h \times N_s}$ denote the snapshot matrix collecting all the N_s snapshot vectors as columns.

Using the POD, r basis vectors are obtained and collected in a matrix $\boldsymbol{V}_r \in \mathbb{R}^{N_h \times r}$, whose i-th column, i.e., the i-th basis vector, corresponds to the i-th eigenvalue $\lambda^{(i)}$ of the correlation matrix $\boldsymbol{C} = \boldsymbol{S}^{\mathrm{T}}\boldsymbol{S}$, $1 \le i \le r$. In fact, given the SVD of the snapshot matrix $\boldsymbol{S}$, written as

$$\boldsymbol{S} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{Z}^{\mathrm{T}}, \tag{18}$$

the basis vectors in $\boldsymbol{V}_r$ are the first r columns of $\boldsymbol{U}$, i.e., $\boldsymbol{V}_r = \boldsymbol{U}[:, 0 : r-1]$ in a Python notation, and $\boldsymbol{\Sigma}$ is a diagonal matrix of singular values, i.e., $\boldsymbol{\Sigma} = \text{diag}(\sqrt{\lambda^{(1)}}, \sqrt{\lambda^{(2)}}, \ldots, \sqrt{\lambda^{(N_s)}})$. Especially when the singular values decay fast, a small number of basis vectors can achieve a small projection error according to (16).

In this way, a reduced basis $\boldsymbol{V}_r$ is obtained, reducing the N_h-dimensional, full-order solution space $\mathbb{R}^{N_h}$ to an r-dimensional, reduced space $\text{Col}(\boldsymbol{V}_r)$, Col representing the column space. With a rapid decay of singular values, the dimensionality reduction is significant ($r \ll N_h$) but the accuracy is still under control.

3.2 Reduced-Order Solutions

The discrete solution is approximated as a linear combination of RB vectors $\boldsymbol{V}_r$, written as $\boldsymbol{u}_h \approx \boldsymbol{V}_r \boldsymbol{q}_r$ with $\boldsymbol{q}_r \in \mathbb{R}^r$ denoting the RB coefficients, and project the full-order system (5) onto the reduced space $\mathrm{Col}(\boldsymbol{V}_r)$. A reduced-order system is thus obtained as

$$\rho \boldsymbol{M}_r \left[\ddot{\boldsymbol{q}}_r(\boldsymbol{\mu}) + \eta \dot{\boldsymbol{q}}_r(\boldsymbol{\mu})\right] + \boldsymbol{A}_r(\boldsymbol{\mu})\boldsymbol{q}_r(\boldsymbol{\mu}) = h(t; \boldsymbol{\mu})\boldsymbol{f}_r(\boldsymbol{\mu}), \tag{19}$$

in which the reduced-size matrices $\boldsymbol{M}_r \in \mathbb{R}^{r\times r}$, $\boldsymbol{A}_r \in \mathbb{R}^{r\times r}$ and $\boldsymbol{f}_r \in \mathbb{R}^r$ are defined as $\boldsymbol{M}_r = \boldsymbol{V}_r^{\mathrm{T}}\boldsymbol{M}_h\boldsymbol{V}_r$, $\boldsymbol{A}_r = \boldsymbol{V}_r^{\mathrm{T}}\boldsymbol{A}_h\boldsymbol{V}_r$ and $\boldsymbol{f}_r = \boldsymbol{V}_r^{\mathrm{T}}\boldsymbol{f}_h$, respectively. Such an r-dimensional reduced system is solved in the online stage for any new parameter value $\boldsymbol{\mu}$.

If the full-size, parameter-dependent stiffness matrix $\boldsymbol{A}_h(\boldsymbol{\mu})$ and source term vector $\boldsymbol{f}_h(\boldsymbol{\mu})$ can be expressed as a linear combination of parameter-independent matrices/vectors with scalar-valued, parameter-dependent coefficients, often referred to as an *affine* form, i.e., $\boldsymbol{A}_h(\boldsymbol{\mu}) = \sum_j \omega_j^a(\boldsymbol{\mu})\boldsymbol{A}_j$ and $\boldsymbol{f}_h(\boldsymbol{\mu}) = \sum_j \omega_j^f(\boldsymbol{\mu})\boldsymbol{f}_j$, one can evaluate their reduced-size counterparts offline as $\boldsymbol{A}_{r,j} = \boldsymbol{V}_r^{\mathrm{T}}\boldsymbol{A}_j\boldsymbol{V}_r$ and $\boldsymbol{f}_{r,j} = \boldsymbol{V}_r^{\mathrm{T}}\boldsymbol{f}_j$, $j = 1, 2, \ldots$, and the online assembly only requires linear combinations $\boldsymbol{A}_r(\boldsymbol{\mu}) = \sum_j \omega_j^a(\boldsymbol{\mu})\boldsymbol{A}_{r,j}$ and $\boldsymbol{f}_r(\boldsymbol{\mu}) = \sum_j \omega_j^f(\boldsymbol{\mu})\boldsymbol{f}_{r,j}$, respectively. In this case, the online assembly is conducted in the reduced dimensionality and guarantees a good online efficiency. However, if an affine form of the full-size matrix/vector is not available, one has to recall the them during the reduced-size assembly, which can often compromise the online efficiency. To overcome the difficulties stemming from the non-affinity, hyper-reduction techniques have been developed to recover an affine approximation of the non-affine operators, see [18, 19] for example.

An alternative approach to recover reduced-order solutions is through non-intrusive surrogate modeling. In addition to the construction of an RB space using the POD, one has to train a regression model to approximate $\boldsymbol{q}_r :]0, T] \times \mathcal{P} \rightarrow \mathbb{R}^r$, $(t, \boldsymbol{\mu}) \mapsto \boldsymbol{V}_r^{\mathrm{T}}\boldsymbol{u}_h(t; \boldsymbol{\mu})$, mapping the time-parameter inputs to the projection coefficients onto the RB. The training data of input–output pairs are derived from a set of collected full-order snapshots. Gaussian process regression has been used for the non-intrusive, reduced-order surrogate modeling in [20–23].

4 Automatic Anomaly Detection with Unbalanced Datasets

This section presents a data-driven technique to detect, localize, and estimate the severity of structural anomalies by observing healthy configurations only. One-class classification learning methods offer the possibility of training a set of samples all belonging to the same *class* and test if a new sample is abnormal, i.e., it belongs to a different class with respect to the training data. Typical one-class classification methods, sometimes called semi-supervised methods, include one-class support vector machines (oc-SVMs), isolation forests, and local outlier factors. During the offline phase, these methods learn a description of the salient features that the training data have in common to ultimately detect if a previously unseen object reflects this description by means of an online anomaly (or novelty) score. If the new unseen sample is associated with an anomaly score close to the ones observed in the training phase, the new object is classified as healthy, otherwise it is classified as damaged. The crucial part is to define what *close to* means from a mathematical standpoint. Let x be an unseen object, then the outcome of all one-class classification methods can be summarized as follows:

$$\begin{cases} \text{score}(x) \geq \theta & \text{damaged/outlier} \\ \text{score}(x) < \theta & \text{healthy/inlier} \end{cases}, \tag{20}$$

where $\text{score}(x)$ is the anomaly score associated with x and θ is an ad hoc threshold to be estimated by observing the anomaly score value of healthy data only. From a practical perspective, in the semi-supervised context, θ is heuristically chosen by observing the highest anomaly score value in the training data, i.e., θ should be equal to the anomaly score of the most outlier sample among all the inliers. Consider $\mathcal{D}$ the dataset of healthy measurements and $\varepsilon \in \mathbb{R}$, then the threshold value is fixed as

$$\theta = \max_{x \in \mathcal{D}} \text{score}(x) + \varepsilon, \tag{21}$$

where a positive value of ε indicates the user accepts a higher false alarm rate, while a negative value implies a higher miss detection rate. The trade-off between false positive and false negative errors should guide the choice of ε and ultimately of θ. It becomes clear that to choose an effective threshold value, the training set has to be the most comprehensive as possible, covering several healthy environmental and operational scenarios.

An alternative approach to detect anomalies is to include sensor measurements belonging to damaged structures in the training set, which leads to using traditional two-class supervised learning methods to distinguish healthy scenarios from damaged ones. In this approach, the choice of the threshold value benefits form the availability of two (or more) classes in the training phase. However, an increasing trend toward the assumption that it would be unreasonable to describe all types of damages is observed in the literature; as a consequence, representing only some damaged configurations would lead to a bias toward certain types and therefore to possible misdetections with high probability (see, e.g., [5, 24]). For this reason, in this chapter, one-class classification methods are used instead.

The general one-class approach is introduced in Sect. 4.1 and explain the need for feature selection in subsection 4.2. For a detailed description of the classic one-class models, the reader is referred to [25, 26] for the oc-SVMs, [27] for the isolation forest and [28] for the local outlier factor. A Python implementation of these methods can be found for example in scikit-learn library [29].

4.1 Local Semi-supervised Method

Considering that the time signals, i.e., the output of interests (2), are collected at multiple sensor locations, one has to decide on how to best aggregate these data. There exists two typical approaches in the literature to combine sensor data: decision-level fusion or feature-level fusion. The latter combines data after feature extraction and considers one global classifier (sensor independent), thus exploiting the correlations across sensors. On the other hand, in decision-level fusion, the signals are classified for each sensor location by a local classifier (sensor dependent) and the results are then combined into a decision output. The two strategies are summarized in Fig. 1. While the superiority of one method over the other one depends strongly on the problem at hand, to exploit the local aspect of the data the authors propose to use the decision-level fusion approach which facilitates the use of a hierarchical classification approach where increasing levels of damage identification can be defined to ultimately gain information on the existence, localization, and severity of the damage.

In a decision-level fusion approach, one has to train as many one-class algorithms as the number of sensor locations. Thus, the global classification model (20) is replaced with m local detection models, where m is the number of sensors:

$$\begin{cases} \text{score}_i(x) \geq \theta_i & \text{damage in the proximity of the } i^{th} \text{ sensor}, \\ \text{score}_i(x) < \theta_i & \text{health in the proximity of the } i^{th} \text{ sensor}, \end{cases} , \tag{22}$$

for $1 \leq i \leq m$. From a computational cost point of view, note that the process can be run in parallel since the local models are independent. Moreover, in the feature-based fusion approach, aggregating the local features leads to high-dimensional input data, while the dimensionality of the input data for the classifiers in the decision-based

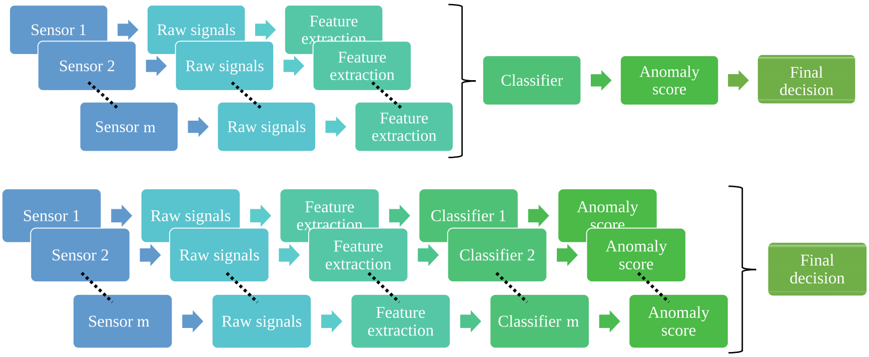

Fig. 1 Flowchart to compare the feature-level (*top*) and the decision-level (*bottom*) fusion approaches for the semi-supervised damage detection strategy with multidimensional training data captured by m sensors

strategy remains small. As further explained in Sect. 4.2, high-dimensional data may lead to overfitting, a well-known problem in machine learning.

4.2 Damage-Sensitive Features

Overfitting refers to the phenomenon observed when the model performs well on training data, but fails to generalize well to new observation. An overfitted model is described by more parameters than can be justified by the data and is typically associated with high-dimensional input data. While sometimes adding more training samples may be a remedy to overfitting, a more effective solution to prevent this behavior is to reduce the dimensionality of the data, i.e., to perform what is called *feature selection*.

Recall that the input data correspond to N_t-dimensional sensor signals, with N_t the number of time steps, which may be very large. Therefore, one needs to express these high-dimensional time signals by means of few variables, extracted from the signals themselves. Ideal features should be sensitive to damage and, at the same time, robust toward noise and healthy variations. Common choices for the engineering-based, damage-sensitive features can be found for example in [24, 30]. When studying the acoustic-elastic equation, i.e., in the context of guided-waves, relevant features are the crest factor, which indicates how extreme the peaks are in a waveform, the maximum and minimum values of the time response, the corresponding arrival times, i.e., the onset, and the number of peaks and troughs in the signals. Without further detail, the reader is referred to [5] and references therein for a thorough description of damage-sensitive features for a guided-wave monitoring approach.

Finally, note that autoencoders, a particular type of neural networks, trained to attempt to copy their inputs to their outputs, have gained particular interest in the framework of anomaly detection, see, e.g., [31–33]. The main advantage of using autoencoders for anomaly detection is that specific engineering-based damage indicator features do not need to be specified by the user, different from classic one-class methods. Instead, by learning the features which suffice to describe and reconstruct the input, autoencoders provide a purely data-driven feature extraction method. Hence, raw measurements such as sensor time signals can be used directly.

5 Finding Optimal Sensor Locations Using Gaussian Processes

Until this point, this work has covered the problem of how to best detect damages given a fixed network of sensors. The specular research question is how to choose the location (and the number) of sensors in order to best detect defaults. Sensor placement strategies are extremely important to optimally equip structures, whose monitoring performance depends critically on the quality of the information collected by sensors. Hence, it is no surprise that this problem has been extensively addressed in the SHM literature, see, e.g., the thorough review [34] and references therein. For most of the sensor placement strategies, the objective is to optimize a suitable cost function with respect to some operational parameters, e.g., the candidate sensor locations and the available number of sensors. However, classic cost functions are usually formulated in terms of damage detectability, which poses a problem when one wishes to make no assumption on the potential damages. Thus, a procedure to place a fixed budget of sensors in the context of anomaly detection is proposed, i.e., when only healthy scenarios are included in the training phase. The proposed strategy relies on sparse Gaussian process to identify the spacial positions that minimize the reconstruction error of an output of interest at all "unsensed" locations. The quantity of interest that defines the cost function for the sensor placement optimization algorithm is the same quantity used to train the anomaly detection classifier, i.e., the damage-sensitive features extracted from the synthetic sensor measurements (11), as described in Sect. 4. As such, the proposed placement strategy is based on an appropriate indicator of the damage detection performance of a given network. Note that, while this approach requires the number and type of sensors to be fixed, it can be easily extended to help the user to identify the minimum number of sensors to achieve a preset coverage.

This section presents the sensor placement strategy introduced in [35], to which the reader is referred for a more in-depth discussion. In particular, after a brief introduction to Gaussian process (GP) regression and sparse GP in Sect. 5.1, the description of how to leverage this technique to systematically place sensors on a structure of interest is provided in Sect. 5.2.

5.1 *(Sparse) Gaussian Process Regression*

A GP is a collection of random variables, any finite number of which obeys a joint Gaussian distribution. In Gaussian process regression (GPR), the prior of regression function is assumed to be a GP corrupted by an independent Gaussian noise term, i.e., for $(\boldsymbol{x}, \boldsymbol{x}') \in \Omega \times \Omega$ with $\Omega \subset \mathbb{R}^{d_\Omega}$ denoting the domain of regression,[1]

$$f(\boldsymbol{x}) \sim \mathrm{GP}(\boldsymbol{0}, \kappa(\boldsymbol{x}, \boldsymbol{x}')), \quad y = f(\boldsymbol{x}) + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \chi^2). \tag{23}$$

There are many different options for the covariance/kernel function $\kappa : \Omega \times \Omega \to \mathbb{R}$, a typical form of which is written as

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \sigma^2 \phi(r), \tag{24}$$

[1] Here f denotes a generic regression function and should not be confused with the linear functional $f(\cdot; \boldsymbol{\mu})$ in Sect. 2.

where $\phi(\cdot)$ is a radial basis function and r [2] can be defined as

$$r = \|\boldsymbol{x} - \boldsymbol{x}'\|/\ell \quad \text{or} \quad r = \sqrt{\sum_{k=1}^{d_\Omega} \frac{(x_k - x_k')^2}{\ell_k^2}},$$

the former for a stationary kernel with isotropic lengthscale ℓ, while the latter for an automatic relevance determination (ARD) kernel that considers an individual correlated lengthscale ℓ_k for each input dimension and allows for differentiated relevances of input features to the regression.

Given a finite number of training input locations in the domain Ω, a prior joint Gaussian is defined for the corresponding outputs:

$$\boldsymbol{y}|\boldsymbol{X} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{K}_y), \quad \boldsymbol{K}_y = \text{cov}[\boldsymbol{y}|\boldsymbol{X}] = \kappa(\boldsymbol{X}, \boldsymbol{X}) + \chi^2 \boldsymbol{I}_M, \tag{25}$$

where $\boldsymbol{y} = \{y_1, y_2, \ldots, y_M\}^{\mathrm{T}}$, $\boldsymbol{X} = [\boldsymbol{x}_1|\boldsymbol{x}_2|\cdots|\boldsymbol{x}_M]^{\mathrm{T}}$, $\boldsymbol{I}_M$ is the M-dimensional unit matrix, and M is the number of training samples.

The goal of a regression model is to predict the noise-free output $f^*(\boldsymbol{s})$ at any new, unseen input location $\boldsymbol{s} \in \Omega$. By the standard Bayesian rule $p(f^*(\boldsymbol{s})|\boldsymbol{X}, \boldsymbol{y}) = p(f^*, \boldsymbol{y}|\boldsymbol{s}, \boldsymbol{X})/p(\boldsymbol{y}|\boldsymbol{X})$, the posterior distribution conditioned on the training data $(\boldsymbol{X}, \boldsymbol{y})$ can be obtained as a new GP:

$$\begin{aligned} f^*(\boldsymbol{s})|\boldsymbol{X}, \boldsymbol{y} &\sim \text{GP}(m^*(\boldsymbol{s}), c^*(\boldsymbol{s}, \boldsymbol{s}')), \\ m^*(\boldsymbol{s}) =& \kappa(\boldsymbol{s}, \boldsymbol{X})\boldsymbol{K}_y^{-1}\boldsymbol{y}, \quad c^*(\boldsymbol{s}, \boldsymbol{s}') = \kappa(\boldsymbol{s}, \boldsymbol{s}') - \kappa(\boldsymbol{s}, \boldsymbol{X})\boldsymbol{K}_y^{-1}\kappa(\boldsymbol{X}, \boldsymbol{s}'), \end{aligned} \tag{26}$$

The values of hyperparameters $\boldsymbol{\theta} = \{\ell \text{or}(\ell_1, \ldots, \ell_d), \sigma^2, \chi^2\}$ make significant difference on the predictive performance. In this chapter, an empirical Bayesian approach of maximizing marginal likelihood is adopted to determine a set of optimal values of the parameters. Using a standard gradient-based optimizer, the optimal hyperparameters $\boldsymbol{\theta}^*$ can be estimated via the maximization problem as follows:

$$\begin{aligned} \boldsymbol{\theta}^* =& \arg\max_{\boldsymbol{\theta}} \log p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta}) = \arg\max_{\boldsymbol{\theta}} \log\left[\mathcal{N}(\boldsymbol{y}|\boldsymbol{0}, \boldsymbol{K}_y(\boldsymbol{\theta}))\right] \\ =& \arg\max_{\boldsymbol{\theta}} \left\{-\frac{1}{2}\boldsymbol{y}^{\mathrm{T}}\boldsymbol{K}_y^{-1}(\boldsymbol{\theta})\boldsymbol{y} - \frac{1}{2}\log\left|\boldsymbol{K}_y(\boldsymbol{\theta})\right| - \frac{M}{2}\log(2\pi)\right\}, \end{aligned} \tag{27}$$

where $p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta})$ is the density function of $\boldsymbol{y}$ given $\boldsymbol{X}$ under hyperparameters $\boldsymbol{\theta}$, considered as the marginal likelihood $p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta}) = \int p(\boldsymbol{y}|\boldsymbol{f}, \boldsymbol{X}, \boldsymbol{\theta})p(\boldsymbol{f}|\boldsymbol{X}, \boldsymbol{\theta})\boldsymbol{f}$.

It is important to remark that the computational complexity of generating a GP model is $\mathcal{O}(M^3)$ and the associated storage requirement $\mathcal{O}(M^2)$, which becomes intractable for large datasets. To overcome the computational limitation, the corresponding sparse methods rely on a small set of $m \ll M$ points, called *inducing*

[2] Here r denotes a (scaled) radius and should not be confused with the reduced dimensionality in Sect. 3.

points, to facilitate the information gain of the whole dataset, thus allowing for a complexity reduction, i.e., $\mathcal{O}(Mm^2)$. An overview of well-known sparse GP regression methods can be found for example in [36], where each sparse method is described as an exact inference with a specific approximated prior, different from the true GP prior (23). A different approach is presented in [37], where both the m inducing points, indicated as $\boldsymbol{D} = [\boldsymbol{d}_1|\boldsymbol{d}_1|\cdots|\boldsymbol{d}_m]^{\mathrm{T}}$, and the hyperparameters $\boldsymbol{\theta}$ are considered as variational parameters to be estimated by minimizing the Kullback–Leibler (KL) divergence between the true posterior (26) and a variational posterior. This is equivalent to maximize the following variational lower bound:

$$\begin{aligned}(\boldsymbol{D}^*, \boldsymbol{\theta}^*) &= \arg\max_{\boldsymbol{D},\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{D}, \boldsymbol{\theta}) \\ &= \arg\max_{\boldsymbol{D},\boldsymbol{\theta}} \left\{\log\left[\mathcal{N}(\boldsymbol{y}|\boldsymbol{0},\, \boldsymbol{Q}(\boldsymbol{D}, \boldsymbol{\theta}) + \chi^2\boldsymbol{I}_M)\right] - \frac{1}{2\chi^2}\mathrm{Tr}(\kappa(\boldsymbol{X}, \boldsymbol{X}) - \boldsymbol{Q}(\boldsymbol{D}, \boldsymbol{\theta}))\right\},\end{aligned} \tag{28}$$

where $\boldsymbol{Q} = \kappa(\boldsymbol{X}, \boldsymbol{D})(\kappa(\boldsymbol{D}, \boldsymbol{D}))^{-1}\kappa(\boldsymbol{D}, \boldsymbol{X})$ is the Nystrom approximation of the true prior covariance. Note that the trace term in (28) acts as a regularization term of the marginal log likelihood, which can be viewed as an accuracy indicator of how well the inducing points summarize the overall statistics.

5.2 Variational Approximation for Systematic Sensor Placement

The aforementioned variational sparse GP model together with the numerical approach defined in the previous sections can be used to systematically place a network of sensors on a structure of interest. Following the description in Sect. 4.2, let $\boldsymbol{y} = \{y_1, \ldots, y_{n_{\mathrm{dof}}}\}^{\mathrm{T}}$ be the damage-sensitive features extracted from the synthetic time signals (11), collected at the n_{dof} points of a coarse mesh of the input domain Ω, which is denoted as $\boldsymbol{X} = [\boldsymbol{x}_1|\boldsymbol{x}_2|\cdots|\boldsymbol{x}_{n_{\mathrm{dof}}}]^{\mathrm{T}}$, where $n_{\mathrm{dof}} \ll N_h$. Given m the number of sensors that the user wishes to place on the structure, one can apply the variational sparse GP strategy to this collection of data, with m being the number of desired inducing points. Ultimately, one can identify the sought sensor locations with the inducing points obtained by variational inference.

Although the procedure is quite simple, some remarks ought to be made. First of all, observe that the hyperparameters and the inducing inputs are estimated by maximizing the variational lower bound (28), which is in general an unconstrained, non-convex optimization problem. This may be problematic because one needs to impose some locality constraints on the inducing points to prevent them from being outside the input domain, especially when this is non-convex. Therefore, the standard variational approximation should be replaced with a constrained optimization:

$$(\boldsymbol{D}^*, \boldsymbol{\theta}^*) = \arg\max_{\boldsymbol{D}\in\Omega_s,\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{D}, \boldsymbol{\theta}), \tag{29}$$

where $\Omega_s \subset \Omega$ indicates the admissible domain for sensor locations and, with a slight abuse of notation, $\boldsymbol{D} \in \Omega_s$ means that each inducing point $\boldsymbol{d}_i$, $1 \leq i \leq m$ is constrained to belong to Ω_s. For real-world problems, the complexity of the domain may be such that the boundaries of Ω_s cannot be easily specified analytically and, in such cases, it may be worth to replace Ω_s with a discrete counterpart. If that is the case, instead of gradient-based optimization techniques, one could opt for discrete optimization methods such as the genetic algorithm [38].

A second point to notice is that the output of interest $\boldsymbol{y}$ is in general parameter dependent, i.e., $\boldsymbol{y} = \boldsymbol{y}(\boldsymbol{\mu})$. Hence, choosing the sensor locations as the optimal inducing points obtained for one specific input configuration may not be optimal for another context, described by a different parameter. To overcome this, this work proposes to apply the variational sparse GP approach to N_μ outputs of interest $\boldsymbol{y}(\boldsymbol{\mu}_i)$, with $\boldsymbol{\mu}_i \in \mathcal{P}$ for $1 \leq i \leq N_\mu$. To summarize the information from the so-obtained $N_\mu m$ inducing points, the K-medoids algorithm, a well-known unsupervised clustering technique, is employed to find m clusters and their corresponding centers, called centroids. As a last step, the desired sensor locations will be chosen as the clusters' centers.

To quantify the quality of the placements, the simplest choice is to compare the relative reconstruction error of the high-fidelity quantity of interest at unsensed locations with respect to the mean of the posterior distribution of the sparse model based on the estimated variational parameters. Alternatively, the pointwise relative variance reduction, defined as

$$V_i = \frac{\kappa(\boldsymbol{x}_i, \boldsymbol{D}^*)(\kappa(\boldsymbol{D}^*, \boldsymbol{D}^*))^{-1}\kappa(\boldsymbol{D}^*, \boldsymbol{x}_i)}{\kappa(\boldsymbol{x}_i, \boldsymbol{x}_i)}, \quad \text{for } 1 \leq i \leq n_{\text{dof}}, \tag{30}$$

provides an indicator on how much variance reduction can be achieved by including $\boldsymbol{x}_i$ to the set of selected sensor locations. When the relative variance reduction is close to one, it means that the inducing variables alone can well reproduce the full GP prediction.

6 Numerical Example

In this section, a numerical problem in 2D is used to illustrate the results in terms of damage detection and sensor placement. Similar results for more complex 3D problems can be found in [5, 35].

Damage detection

The first step consists in generating a synthetic database of sensor observations. Apply (1) with homogeneous free slip boundary conditions and homogeneous initial conditions, i.e., $\boldsymbol{u}_0 = \boldsymbol{v}_0 = \boldsymbol{t}_N = 0$, to the healthy geometry illustrated in Fig. 2a and equipped with $m = 15$ sensors. The high-fidelity numerical solutions are computed using the FE approximation with $\mathbb{P}_1$ elements over a discretized domain with

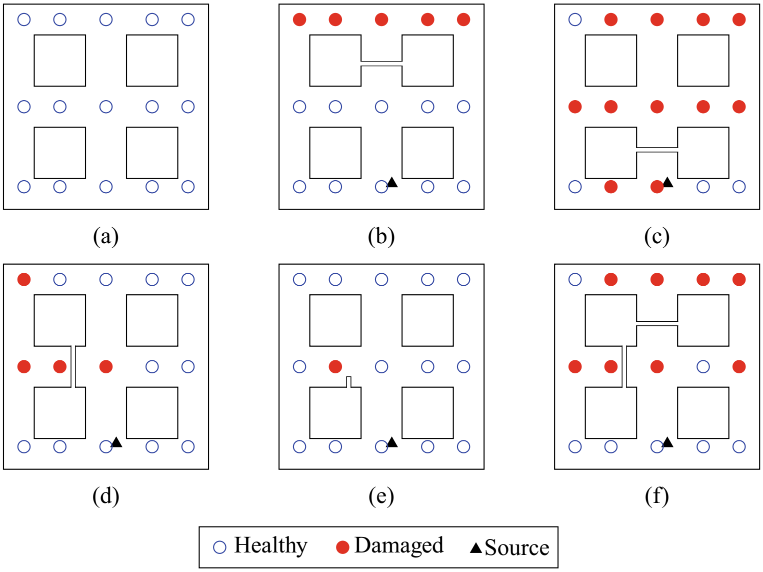

Fig. 2 Summary of the one-class SVMs classification results for one healthy geometry (**a**) and 5 damaged ones (**b**–**f**)

a total of $N_h = 30'912$ degrees of freedom, while for the RB solver the model relies on 267 basis. The natural variations are described by $d_\mu = 3$ parameters, i.e., $\boldsymbol{\mu} = (E, \nu, k) \in \mathcal{P} = [0.999, 1.001] \times [0.329, 0.331] \times [1.9, 2.1]$, where E is the Young's modulus and ν the Poisson's ratio, defining the stress tensor $\boldsymbol{\sigma}$, while k is a parameter representing the number of cycles before attenuation of the source impulse. The position of the active source as well as the density and damping coefficients are fixed, i.e., $\rho = 1$, $\eta = 0.1$, respectively.

For each sensor location, one should follow the pipeline introduced in Sect. 4 and generate a training dataset of $N_{\text{train}} = 1000$ samples, obtained by extracting a few damage-sensitive features from the discrete time signal (11), in their turn obtained with a suitable reduced-order modeling approach, as described in Sect. 3. Then, one has to train m separate one-class SVM models to learn the common traits of the local healthy features and test the results on sensor measurements belonging to both healthy and damaged geometries. In this example, the test signals are generated synthetically using the high-fidelity model. Different from the training set, to approximate experimental measurements, a Gaussian noise term $\varepsilon_i \sim \mathcal{N}(0, \gamma_i^2)$ is added to the test time signal (11), with γ being the 0.01% of the maximum amplitude of 30 randomly chosen training healthy signals in the training set. Damages are obtained by modifying the geometry of the structure to include discontinuities, as shown in Fig. 2b–f. The crack-modeling approach is a common choice in the literature, see, e.g., [39] where artificial damages on the blade of a wind turbine are implemented via a trailing edge opening. Anomaly detection results are shown in Fig. 2, where for each geometry the average outcome of the oc-SVM for 10 simulations (i.e., for 10 different input parameters) is presented. For each damaged scenario, at least one sensor is classified as damaged, while for the healthy scenario all sensors are classified as healthy. Moreover, for most of the damaged scenarios, one can observe a certain level of proximity between the cracks and the sensors classified as damaged, thus guiding the localization of damages. An exception corresponds to the geometry in Fig. 2c, where almost all sensors are classified as damaged, thus preventing localization. This issue is attributable to the relative position of the source and the crack, i.e., to localize damage (c) the source should be placed differently. A reasonable solution is to consider different locations for the active source, for example by defining an additional input parameter in the model. This approach is already employed in SHM with Lamb wave propagation where piezoelectric transducers are used as both sensors and actuators (see e.g., [40]).

Sensor placement

To identify the optimal sensor locations, one has to create a new database of $N_\mu = 100$ synthetic observations sampled from a Sobol's sequence [41]. Keep the same parameters as in the previous paragraph, but, instead of computing the synthetic time signals at few predefined points in space (the sensor locations), collect the measurements at all the nodes of a new coarse mesh of Ω, for a total of $n_{\text{dof}} = 360$ degrees of freedom. Following the description given in Sect. 5.2, for each one of the N_μ time signals, one has to perform the constrained variational approximation (29) with an ARD-Exponential kernel over $\Omega_s = \Omega \setminus \partial\Omega$ to obtain a set of m inducing points $\boldsymbol{D}^*(\boldsymbol{\mu}_i)$ for $1 \leq i \leq N_\mu$. Figure 3 sketches the clustering results obtained for different values of N_μ. Observe that the centroids tend to stabilize already after considering a cluster of 10 samples, especially when the sensor budget is high, i.e., for large m values. This can be explained by noticing that when m is small, the algorithm is trying to reconstruct a non-trivial quantity of interest over a complex domain with only a few points, which may lead the sparse model to get stuck in a local minimum without reaching convergence. Finally, the relative variance reduction (30) is used to evaluate the quality of the estimated locations whose results are shown in Fig. 4, where one can observe a variance reduction almost equal to 1 near the sensor locations and a general reduction above 0.7 for all the unsensed locations even for $m = 4$, thus indicating a good sensor placement.

7 Conclusion

This chapter presents how a model-based numerical approach can be integrated with different data-driven techniques in the context of predictive maintenance. A peculiarity of this work is that the authors make no assumption on the type of damages that a structure may undergo during its lifetime, while modeling many environmental and operational healthy scenarios. From a technical point of view, this work describes how reduced-order modeling techniques can be leveraged to generate large and robust

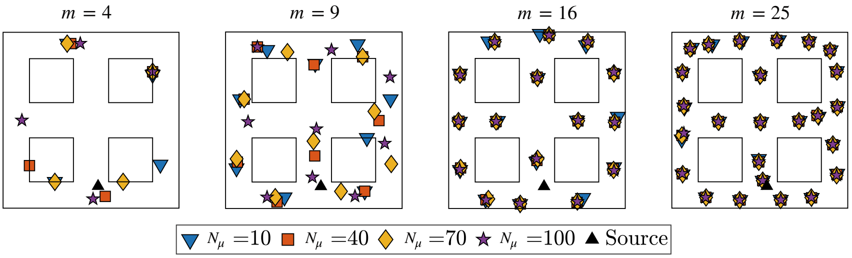

Fig. 3 Comparison of the centroids obtained with the K-medoids clustering algorithm for different sizes of clusters N_μ. Each plot shows a fixed number m of inducing points, which increases from left to right

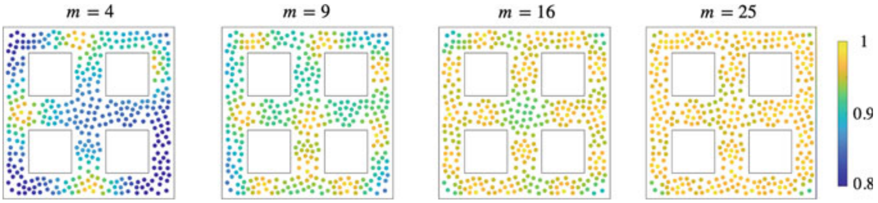

Fig. 4 Relative variance reduction obtained using m centroids and averaged over $N_\mu = 100$ samples. Each plot shows a fixed number m of inducing points, which increases from left to right. Values close to 1 indicate a good placement quality

datasets of synthetic sensor measurements and it explains how such datasets can be used to learn the salient features that healthy scenarios have in common to ultimately detect damages. The same damage-sensitive features are also used to guide an automated data-driven sensor placement strategy to increase the detection accuracy for a given budget of sensors.

Although a simple 2D example is used to validate the proposed method, the generalization to more complex, possibly nonlinear problems is possible. Note however that for real-world engineering problems the parameter space describing healthy variations is expected to be high dimensional, thus requiring a high computational effort to elaborate the synthetic time signals. To this end, sensitivity analysis techniques such as the variance-based global sensitivity indices [42] or the derivative-based global sensitivity measures (DGSM) [43] are two popular choices to identify a few parameters that influence the output of interest the most. Finally, note that while the proposed method can be used for real-time predictions thanks to the offline and online decomposition of tasks, a filtering technique to integrate the evolution of the structure and update the model would be a valuable addition.

References

1. Farrar CR, Worden K (2012) Structural health monitoring: a machine learning perspective. Wiley, New York
2. Wagg DJ, Worden K, Barthorpe RJ et al (2020) Digital twins: state-of-the-art and future directions for modeling and simulation in engineering dynamics applications. ASCE-ASME J Risk Uncertainty Eng Syst Part B: Mech Eng 6(3):030901
3. Lecer M, Allaire D, Willcox K (2015) Methodology for dynamic data-driven online flight capability estimation. AIAA J 53(10):3073–3087
4. Taddei T, Penn JD, Yano M, Patera AT (2018) Simulation-based classification; a model-order-reduction approach for structural health monitoring. Arch Comput Methods Eng 25(1):23–45
5. Bigoni C, Hesthaven JS (2020) Simulation-based anomaly detection and damage localization: an application to structural health monitoring. Comput Methods Appl Mech Eng 363:112896
6. Kapteyn MG, Knezevic DJ, Willcox K (2020) Toward predictive digital twins via component-based reduced-order models and interpretable machine learning. In: AIAA Scitech 2020 Forum, p 0418
7. Rosafalco L, Manzoni A, Mariani S et al (2020) Fully convolutional networks for structural health monitoring through multivariate time series classification. Adv Model Simul Eng Sci 7(1):1–31
8. Quarteroni A, Saleri F, Gervasio P (2006) Scientific computing with MATLAB and Octave. Springer, Berlin
9. Zienkiewicz OC, Taylor RL (2005) The finite element method for solid and structural mechanics. Elsevier, Amsterdam
10. Newmark NM (1959) A method of computation for structural dynamics. J Eng Mech Div 85(3):67–94
11. Hesthaven JS, Rozza G, Stamm B (2016) Certified reduced basis methods for parametrized partial differential equations. Springer, Berlin
12. Quarteroni A, Manzoni A, Negri F (2015) Reduced basis methods for partial differential equations: an introduction. Springer, Berlin
13. Benner P, Gugercin S, Willcox K (2015) A survey of projection-based model reduction methods for parametric dynamical systems. SIAM Rev 57(4):483–531
14. Haasdonk B, Ohlberger M (2008) Reduced basis method for finite volume approximations of parametrized linear evolution equations. ESAIM Math Model Numer Anal 42(2):277–302
15. Wang Q, Ripamonti N, Hesthaven JS (2020) Recurrent neural network closure of parametric POD-Galerkin reduced-order models based on the Mori-Zwanzig formalism. J Comput Phys 410:109402
16. Choi Y, Carlberg K (2019) Space-time least-squares Petrov-Galerkin projection for nonlinear model reduction. SIAM J Sci Comput 41(7):A26–A58
17. Bigoni C (2020) Numerical methods for structural anomaly detection using model order reduction and data-driven techniques. Ph.D. thesis No. 7734, EPFL
18. Barrault M, Maday Y, Nguyen NC, Patera AT (2004) An ‘empirical interpolation’ method: application to efficient reduced-basis discretization of partial differential equations. Comptes Rendus Mathematique 339(9):667–672
19. Chaturantabut S, Sorensen DC (2010) Nonlinear model reduction via discrete empirical interpolation. SIAM J Sci Comput 32(5):2737–2764
20. Kast M, Guo M, Hesthaven JS (2020) A non-intrusive multifidelity method for the reduced order modeling of nonlinear problems. Comput Methods Appl Mech Eng 364:112947
21. Zhang Z, Guo M, Hesthaven JS (2019) Model order reduction for large-scale structures with local nonlinearities. Comput Methods Appl Mech Eng 353:491–515
22. Guo M, Hesthaven JS (2019) Data-driven reduced order modeling for time-dependent problems. Comput Methods Appl Mech Eng 345:75–99
23. Guo M, Hesthaven JS (2018) Reduced order modeling for nonlinear structural analysis using Gaussian process regression. Comput Methods Appl Mech Eng 341:807–826

24. Long J, Buyukozturk O (2014) Automated structural damage detection using one-class machine learning. In: Catbas FN (ed) Dynamics of civil structures, vol 4. Springer, Berlin, pp 117–128
25. Schölkopf B, Williamson RC, Smola AJ et al (2020) Support vector method for novelty detection. Adv Neural Inf Process Syst 12:582–588
26. Cristianini N, Schölkopf B (2002) Support vector machines and kernel methods: the new generation of learning machines. AI Mag 23(3):31
27. Liu FT, Ting KM, Zhou Z-H (2008) Isolation forest. In: 2008 Eighth IEEE international conference on data mining. IEEE, pp 413–422
28. Breunig MM, Kriegel H-P, Ng RT et al. (2000) LOF: identifying density-based local outliers. In: ACM sigmod record, pp 93–104
29. Pedregosa et al (2011) Scikit-learn: machine learning in Python. J Mach Learn Res 12:2825–2830
30. Liu SW, Huang JH, Sung JC (2002) Detection of cracks using neural networks and computational mechanics. Comput Methods Appl Mech Eng 191(25–26):2831–2845
31. Japkowicz N, Myers C, Gluck M (1995) A novelty detection approach to classification. In: International conference on artificial intelligence, vol 1, pp 518–523
32. Marchi E, Vesperini F, Eyben F et al (2015) A novel approach for automatic acoustic novelty detection using a denoising autoencoder with bidirectional LSTM neural networks. In: 2015 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE, pp 1996–2000
33. Pathirage CSN, Li J, Li L et al (2018) Application of deep autoencoder model for structural condition monitoring. J Syst Eng Electron 29(4):873–880
34. Ostachowicz W, Soman R, Malinowski P (2019) Optimization of sensor placement for structural health monitoring: a review. Struct Health Monit 18(3):963–988
35. Bigoni C, Zhang Z, Hesthaven JS (2020) Systematic sensor placement for structural anomaly detection in the absence of damaged states. Comput Methods Appl Mech Eng 371:113315
36. Quiñonero-Candela J, Rasmussen CE (2005) A unifying view of sparse approximate Gaussian process regression. J Mach Learn Res 6:1939–1959
37. Titsias M (2009) Variational learning of inducing variables in sparse Gaussian processes. In: Proceedings of the twelfth international conference on artificial intelligence and statistics, PMLR, vol 5, pp 567–574
38. Davis L (1991) Handbook of genetic algorithms. CumInCAD
39. Avendaño-Valencia LD, Chatzi EN, Tcherniak D (2020) Gaussian process models for mitigation of operational variability in the structural health monitoring of wind turbines. Mech Syst Sig Process 142:106686
40. Swartz RA, Flynn E, Backman D et al (2006) Active piezoelectric sensing for damage identification in honeycomb aluminum panels. In: Proceedings of 24th international modal analysis conference
41. Joe S, Kuo FY (2008) Constructing Sobol sequences with better two-dimensional projections. SIAM J Sci Comput 30(5):2635–2654
42. Sobol IM (2001) Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates. Math Comput Simul 55(1–3):271–280
43. Kucherenko S, Song S (2016) Derivative-based global sensitivity measures and their link with Sobol'sensitivity indices. In: Owen AB, Glynn PW (eds) Monte Carlo and Quasi-Monte Carlo Methods. Springer, Cham, pp 455–469