Politecnico di Torino

Politecnico di Torino

Master of Science in Civil Engineering

Master of Science Thesis

A Bayesian Optimization Approach for Finite Element Model Updating

Thesis supervisors:

Prof. Rosario Ceravolo

Prof. Luca Zanotti Fragonara

Dr. Marco Civera

Candidate:

Davide Raviolo

CRANFIELD UNIVERSITY

&

Politecnico di Torino
Dipartimento di Ingegneria Strutturale, Edile e Geotecnica

Master Thesis

Academic Year 2020 - 2021

Davide Raviolo

A Bayesian Optimization Approach for Finite Element Model Updating

Thesis supervisors: Prof. Rosario Ceravolo
Dr. Luca Zanotti Fragonara
Dr. Marco Civera

October 2021

Abstract

Model Updating (MU) aims at estimating the unknown properties of a physical system of interest from the actual observations. In numerical models, these unknowns are described by the model parameters. Typically, besides plain model calibration purposes, MU procedures are employed for the Non-Destructive Evaluation (NDE) and damage assessment of structures. In this framework, damage can be located and quantified by updating stiffness-related parameters. Indeed, under unchanged operational and environmental conditions, a local reduction of stiffness may denote localized structural damage. For iterative Model Updating methods that make use of a cost function to be minimized, three major critical aspects may compromise the success of the whole updating procedure: the Finite Element (FE) model validity, the reliability of the experimental data, and the complexity of the optimization problem at the computational level. Usually, sophisticated FE models can generate expensive and non-convex cost functions, the minimization of which is a non-trivial task. To deal with such a challenging optimization problem, this work makes use of a Bayesian optimization approach. In this framework, a prior is set over the objective function and then combined with evidence (i.e., observations) to get a posterior function. This enables the intelligent selection of the next point to be sampled from the objective function, taking into account both exploitation and exploration needs, resulting in a very efficient global optimization technique, that is best-suited for minimizing expensive black-box functions. Bayesian optimization is also deemed as a surrogate optimization technique, since the prior, usually a Gaussian Process (GP), can be seen as a probabilistic surrogate model of the underlying objective function.

The performance of this proposed scheme is compared to three well-established global optimization techniques, namely Generalized Pattern Search, Simulated Annealing, and a Genetic Algorithm procedure. This investigation is made by means of three numerical case-studies, allowing inspecting the capabilities of the software under controlled settings. The last case-study, the bell tower of Santa Maria Maggiore Cathedral in

Mirandola (Italy), served also as an experimental case study, based on data from a previous survey, so to evaluate the performance of the proposed technique in a real-case model updating scenario.

Acknowledgments

I would like to express my deepest gratitude to my thesis supervisor Dr. Luca Zanotti Fragonara for his continued support and wise guidance. I am sincerely grateful for his unremitting availability, in spite of his busy working schedule and the difficulties due to the pandemic: this work wouldn't have been possible without his instructive advice. Likewise, I want to thank my co-supervisor Dr. Marco Civera, whose genuine commitment and invaluable feedback were essential for completing my thesis. Additionally, I am truly thankful to Prof. Rosario Ceravolo, for reviewing my work and for introducing me to my advisors, giving me the opportunity to carry out this thesis at Cranfield University.

Finally, a special thanks goes to my beloved family, that always provided me all the support I needed through the years of my education.

Table of Contents

1 Introduction

Finite Element Model Updating (FEMU) is a well-established technique in the framework of structural engineering, representing an effective tool for model calibration, structural identification, and non-destructive damage assessment. As FE models are parametric models, the output response is affected by changes of input parameters. When measurements of the actual physical system response are available, one may want to compare the measured response and the modeled one. The basic idea is to select the right set of input parameters that minimizes the misfit between these two responses. A practical way is to use a system-intrinsic response, being contingent on the model parameters only, such as modal properties. As changes of input parameters may have little impact on the modal response of the system, it is particularly difficult to estimate the right parameters by minimizing the computed and measured response discrepancy. Model updating is therefore an inverse problem, extremely liable to small errors, that takes response data and deduces information about the underlying model (Friswell M. , 2008). Unfortunately, FE model deficiencies and poor reliability of experimental data are both sources of significant amounts of error.

Iterative model updating methods, that make use of a penalty function (which measures the misfits between measurements and predictions) are among the most popular employed techniques (Sehgal & Kumar, 2016). One of the key advantages of this approach is the possibility of choosing physically meaningful parameters, like material properties (density, Young's modulus, Poisson's ratio, etc.) or geometrical features. For example, damage can be located and quantified by updating the elasticity modulus of the material, after having conveniently discretized the structure and under the assumption that a local reduction of stiffness corresponds to localized structural damage (Friswell & Mottershead, 2001). As this method relies on the minimization of a penalty function, one must take proper care of the constrained optimization problem at the computational level. In fact, the following aspects complicate the optimization problem: (1) the updating

process often involves the solution of complex finite element models, requiring great computational effort at each iteration, (2) the optimization ranges about the updating parameters may be particularly large to capture high uncertainties, widening the optimization space, (3) the number of dimensions can be high, as one may need to estimate many parameters, and (4) the problem can be practically close at being ill-conditioned, given the inverse nature of the updating problem. Hence, usually the user must deal with expensive, high-dimensional, and non-convex penalty functions, which minimization is extremely difficult to achieve. Many global optimization techniques have been developed in the last decades. Among the most well-established ones, generalized pattern search, simulated annealing and Genetic Algorithms (the last two are sometimes deemed as "computational intelligence" optimization techniques) have been extensively employed in finite element model updating problems (Marwala, 2010). Although being developed for global search of the optimum, these techniques all share a major drawback: they require a high number of function evaluations to perform well. For what concerns masonry monumental buildings of historical and architectural interest, a state-of-the-art review of updating procedures used for model calibration and damage detection is given by (Atamturktur & Laman, 2012). For some instances of FEMU applications in this field, see (Zanotti Fragonara, Boscato, & Ceravolo, 2017), (Boscato, et al., 2013) who used a generalized pattern search algorithm, and (Bassoli, et al., 2018) who made use of an advanced surrogate-assisted evolutionary algorithm to try to overcome the sampling efficiency problem.

The aim of this work is to use a Bayesian optimization approach to deal with the minimization of insidious penalty functions in the framework of FE model updating problems. Bayesian optimization is a relatively new, yet very powerful optimization technique, which is very efficient at sampling the objective function (a highly desirable quality when facing expensive functions), while showing excellent global search aptitudes (Mockus J. , 1989). This is achieved by taking advantage of a probabilistic surrogate model and "intelligently" selecting the sampling points by means of an acquisition function, that performs an automatic tradeoff between exploitation (sampling from areas expected to offer improvement over the incumbent best observation) and exploration (sampling from areas of high uncertainty). To assess the qualities of the

Bayesian optimization method in structural dynamics applications, its performance is compared with the aforementioned global optimization techniques by means of three case-studies. The first one, a simple shear-type frame, is used to assess the correct functionality of the four algorithms. The second case-study consists of a rather complex frame structure, which is modeled by an expensive FE model. This problem enables to assess the performance of Bayesian optimization and investigate the impact that some implementation choices have on the updating outcome. Finally, the third case-study consists in the damage assessment of an historical building, the bell tower of the Santa Maria Maggiore cathedral in Mirandola, Italy. This model updating setup serves both as a numerical case-study and as an experimental case-study, allowing to evaluate the performance of the Bayesian optimization approach in a real-case model updating scenario.

This work is organized as follows. Firstly, in Section 2, after giving an overview about finite element model updating and some of its most critical aspects, generalized pattern search (GPS), simulated annealing (SA) and the Genetic Algorithm (GA) optimization techniques will be briefly described, followed by a more detailed account of the Bayesian optimization method. In Section 3, further details will be given about the implementation used for the algorithms in the analyzed cases. Subsequently, the three model updating case-studies will be discussed in Section 4, by investigating the behavior of the algorithms and comparing their performance throughout the optimization process. Finally, further developments are discussed in Section 5, which also contains some conclusive thoughts about the results previously obtained.

2 Theoretical background

In many engineering problems use is made of numerical models to coherently represent physical systems. For the numerical model to achieve its purpose, whatever it might be, its validity is of uttermost importance, as the numerical model must be capable of faithfully represent the response of the physical system. In short, model updating techniques aim at fulfilling this need.

The output of such numerical models depends on some initial parameters, which are often unknown or uncertain. In the field of structural dynamic applications, these parameters constitute system properties as, for example, material properties, geometric properties, load conditions and constrain conditions. In this framework, where model updating is also known as *model calibration* or *system identification*, it is also assumed that certain inefficiencies of the numerical (finite element) model to accurately represent the underlying physical system are accounted for by appropriately changing some of the model parameters.

It goes without saying that some sort of measure of the actual system response is needed in model updating, since the process practically attempts to quantify a "degree of correlation" between the measured response and the modeled one. It may sound prosaic, but it's essential not to forget that the identification of the system response (e.g., modal data or frequency response data) may often be a difficult task and source of much uncertainty. Besides experimental measurements errors, a lack of correlation between predictions and observations caused by the following causes of inaccuracy in the numerical model, as well explained in the work of (Mottershead & Friswell, 1993):

- *Model structure errors* – due to occur in the case of uncertain underlying physical equations.
- *Model order errors* – due to occur when a continuous system is inappropriately discretized to an extent the numerical model is no longer able to capture the

system response at the required order (i.e., when a FE model is not able to render the structural behavior at higher modes because the number of DOF is insufficient).

- *Model parameters errors* – due to a lack of knowledge of system properties or inappropriate constraints setup.

Once the model structure and model order have been conveniently taken care of, the problem of *model calibration/system identification* reduces to a matter of parameters estimation. Another well-known application of model updating is *damage detection* in structural health monitoring: a numerical model (eventually previously calibrated) is updated so that areas with local reduction of material stiffness are identified as regions that suffered a certain amount of structural damage.

A model is generally conceived as a set of the relationships between the input and output variables of a system. Parametric models (e.g., finite Elements Models) are described by a vector of model parameters $\boldsymbol{\theta}$. Thus, being M the model operator, $\boldsymbol{y} = \boldsymbol{M}(\boldsymbol{x}, \boldsymbol{\theta})$ returns the output vector $\boldsymbol{y}$ for a given input vector $\boldsymbol{x}$. For obvious reasons, in model updating it is preferable to adopt outputs that are independent on the input and dependent only on the model parameters (e.g., Eigen data): this is indeed the common practice in model updating applications. According to this assumption, the $\boldsymbol{x}$ vector can be dropped, and we may simply write $\boldsymbol{y} = \boldsymbol{M}(\boldsymbol{\theta})$.

Finite elements model updating methods fall in two categories, direct methods and iterative methods (the latter also called deterministic), (Friswell & Mottershead, Finite Element Model Updating in Structural Dynamics, 1995). *Direct methods* try to improve observed data and computed data agreement by directly changing the mass and stiffness matrices; this leads to little physical meaning (no correlation with physical model parameters), problems with elements connectivity and fully populated stiffness matrices. For these reasons, they are seldomly used in common structural engineering applications. The *iterative methods* attempt to obtain results that fit the observations by iteratively changing the model parameters: this enables to retain good physical understanding of the model and doesn't present the above-mentioned problems. Consequently, this study makes used of iterative methods, as these are usually preferred in engineering

applications. The degree of correlation is determined by a *penalty function* (or *cost function*): optimizing this function requires the problem to be solved iteratively, which means computing the output (i.e., performing a FE analysis) of the numerical model at each iteration. Hence, a greater computational cost of the updating process is the major drawback of the iterative methods.

As described, model updating is an inverse problem, since it aims at inverting the relationship between the model parameters and the model output to find the *optimal* set of parameters $\boldsymbol{\theta}$ that minimizes the misfit between computed data and measured data. In this sense, model updating can be simply considered as the following constrained optimization problem:

$$\boldsymbol{\theta}^* = \arg\min_{\theta \in D} \mathrm{F}(M(\boldsymbol{\theta}), \boldsymbol{f}) ,$$

where $\boldsymbol{\theta}^*$ is the set of optimal parameters, D is the parameter space, F is the cost function and $\boldsymbol{f}$ is the measured data.

The whole process of solving $\mathrm{F}(M(\boldsymbol{\theta}), \boldsymbol{f})$ – the output of the numerical model "post-processed" in some way by the penalty function – may be conceived as computing an unknown (non-linear) objective function of the model parameters $\boldsymbol{\theta}$, which constitute the de-facto input of the numerical model to be updated. Typically, this objective function is non-convex and expensive to evaluate. The output surface of the objective function lies in a d −dimensional space, where d is the number of the parameters to be optimized. The sampling volume is exponential to d (using a spacing of 10^{-n} for each dimension, the number of sampling points for a d −dimensional hypercube is 10^{n^d}), thus posing an implicit (and sometimes despicable) restriction to the number of parameters that can be optimized without incurring in computational problems: this is known as the "course of dimensionality". Moreover, if it's true that computers' computational power has always been increasing, on the other hand also numerical models (this is for sure the case for finite elements models in structural mechanics applications) are becoming more and more complex, and so more computationally demanding. The need for very efficient optimization techniques suitable for potentially highly non-linear black-box functions is therefore clear. Nonetheless, an optimization algorithm should be able to identify the

global minimum across the function domain, avoiding running into local minima. Unfortunately, sampling efficiency and global search aptitudes are somewhat conflicting goals.

Many optimization algorithms have been developed in the last decades, each of them with their peculiar strengths and weaknesses. Among them, three of the better known and most extensively used are the GPS algorithm - which does not require the computation of the gradient or the derivatives of the function -, Genetic Algorithms (GA) and simulated annealing algorithms (SA) - that are both stochastic/heuristic search algorithms.

During the last two decades, and last years in particular, Bayesian optimization has proven itself to be a powerful strategy for finding the global minimum of non-linear functions that are expensive to evaluate, non-convex and whose access to the derivatives is burdensome[1]. Furthermore, the Bayesian optimization technique distinguishes itself for being one of the most efficient approaches in terms of the number of objective function evaluations (Mockus J. , Application of Bayesian approach to numerical methods of global and stochastic optimization., 1994), (Jones, Schonlau, & Welch, 1998), (Streltsov & Vakili, 1999), (Jones, 2001), (Sasena, 2002).

The essence of the Bayesian optimization lies in the reading of the optimization problem given by the "Bayes' Theorem" (hence the name Bayesian):

$$P(M|E) \propto P(E|M)\, P(M)\,,$$

which mathematically states that the conditional probability of event M occurring given the event E is true is proportional to the conditional probability of event E occurring if event M is true multiplied by the probability of M. Here, $P(M|E)$ is seen as the *posterior probability* of the model M given the evidence (or observations) E, $P(E|M)$ as the *likelihood* of E given M and $P(M)$ as the *prior probability* of the model M. Essentially, the prior, $P(M)$, represents the extant beliefs about the type of possible objective functions, given some knowledge we have on some function properties based on the observations already at our disposal. The posterior $P(M|E)$, on the other hand, represents

[1] Iterative methods often generate non-smooth penalty functions that make the calculation of the gradient extremely difficult and sometimes impossible due to badly scaled matrices (Marwala, 2010).

our updated beliefs about the objective function, given the likelihood of the new observations we just made. Since conjectures are made about the objective function (that is, the output of the numerical model post-processed by the cost function), the process basically aims at its estimation by means of a surrogate function, or *surrogate model.*

Many stochastic regression models can be used as surrogate models: the model must be able to describe a predictive distribution that represents the uncertainty in the reconstruction of the objective function, in practice by providing a mean and a variance. Gaussian Processes are by far the most used surrogates in Bayesian optimization applications, but also other probabilistic regression models, like Random forests models, have recently gained in adoption.

To efficiently select the next point to be sampled, that is the next point the objective function is evaluated in, Bayesian optimization techniques make use of an *acquisition function* defined over the moments of the posterior distribution given by the surrogate model (e.g., a GP). The role of the acquisition function is crucial, since it governs the trade-off between exploration (aptitude for the global search of the minimum) and exploitation (aptitude to sample regions where the function is expected to be low) of the optimization process. Probability of improvement (PI), expected improvement (EI) and upper confidence bound (UCB) are among the most used and most popular acquisition functions in Bayesian optimization applications.

In this section, the topics just introduced are formally presented, and a theoretical, more detailed, perspective of these processes and some of the related caveats is given. First, in Paragraph 2.1, the problem of finite element model updating and some of its (problematic) aspects (namely ill-posedness, parameters selection and ill-conditioning) are further discussed. A brief formal description of the generalized pattern search (GPS) algorithm, the Genetic Algorithm and the simulated annealing algorithm is presented in Paragraph 0. Finally, a theoretical overview of the Bayesian optimization approach is given in Paragraph 2.3: surrogate models, particularly Gaussian Process priors, are covered in Paragraph 2.3.2, while the formal description as well as the strengths and weaknesses of the aforementioned acquisition functions are treated in Paragraph 2.3.3.

2.1 Finite element model updating for structural dynamics applications

When the model to be updated is a finite element one, as may often be the case in common engineering applications, the updating procedure is referred to as finite element model updating. The principles outlined previously fully apply in this case, since finite elements models are in fact parametric models, which output, in the field of structural dynamics, can be used to extract modal/frequency data or to predict the time/frequency response of a structure. In the former case, the output will be dependent on the model parameters only.

In this context, model updating is used for system identification, model calibration and damage detection: in all these applications, it is necessary to retain a high level of physical meaningfulness. Hence, deterministic approaches are by far preferred over direct approaches. In this dissertation, focus will be placed on deterministic approaches.

Many finite element model updating methods have been proposed and successfully used: sensitivity-based methods, (Fox & Kapoor, 1968), (Chen & Garba, 1980), and (Alvin, 1996); eigenstructure-assignment methods, (Zimmerman & M., 1992) and (Biswa, 2002); uncertainty quantification methods (Simoen, De Roeck, & Lombaert, 2015); sensitivity-independent iterative methods, (Levin & Lieven, 1997); and many more, (Wang, Tan, Li, & Liu, 2013).

Giving a through overview of the advantages and disadvantages of each of the above methods is not the purpose of this dissertation. Nonetheless, to better understand the reasoning behind the sensitivity-independent iterative optimization methods and their benefits, some of the shortcoming of the sensitivity-based methods (arguably one of the most straightforward and well-established updating techniques) may be summarized as follows (Marwala, 2010):

- Derivatives are computed at a local level: this leads to high chances of getting stuck in a local minimum of the objective function.
- The computation of the sensitivity matrix likely causes inefficiencies in high-dimensional problems.

- Results depend upon the set of parameters used to initialize the updating procedure.
- The search of the optimal parameters must occur in a tight range (i.e., a low prior uncertainty on the parameters is required).

For these reasons, it is clear how sensitivity-based methods may be ineffective when attempting to update complex finite element models, which solution leads to rather expensive, high-dimensional, non-linear, and non-convex objective functions.

Iterative optimization methods that don't rely on the computation of the derivatives, and especially the so-called computational intelligence ones, try to overcome the issues highlighted above. The calculation of the gradients, which is needed in traditional optimization techniques, usually results to be an expensive task that may also cause numerical problems such as matrix singularity (Marwala, 2010). While for obvious reasons the peculiar method's characteristics (e.g., sampling efficiency, algorithm efficiency, dimension scalability, exploitation-exploration trade-off, etc.) depend on the type of optimization algorithm employed, treating the objective as a black-box function generally leads to a reduced capability of detecting ill-posedness problems due to an unwise choice of the parameters to be updated. Moreover, discarding gradient information may result in a greater sampling demand in cases where derivatives can be easy to access: optimization efficiency is a major point of concern for these updating methods. Finally, the need arises of tailoring the choice of the optimization technique to the specifics of the updating problem: more often than not, for computational intelligence optimization methods the overall cost-effectiveness of the updating process is a trade-off between the algorithm efficiency and the sampling efficiency. Therefore, the choice of the right optimization technique should be made based on the computational cost of the objective function to be optimized.

To measure the misfit between the measured response and the computed response, modal domain data or frequency domain data (both only dependent on the model parameters) are commonly used. In this work, modal proprieties have been chosen to evaluate the degree of correlation of experimental and theoretical results. Typically, when Eigen data is used in model updating, both the natural frequencies and the corresponding mode

shapes of the dynamic system are used. While comparison between natural frequencies is straightforward, the comparison of mode shapes (that are formally described by vectors) is not as trivial. This is usually achieved using the Modal Assurance Criterion (MAC), that is a statistical indicator capable of measuring the coherence of two Eigenvectors, defined as follows:

$$MAC_{cdr} = \frac{|\{\phi_{cr}\}\{\phi_{dr}^*\}|^2}{\{\phi_{cr}\}^T\{\phi_{cr}^*\}\{\phi_{dr}\}^T\{\phi_{dr}^*\}},$$

where $\{\phi\}$ are the mode shapes to be compared (c denotes the reference, d the degree-of-freedom, r the mode, and $*$ the complex conjugate). MAC values close to 1 stand for modes with high correlation, MAC values close to 0 suggest a low correlation between two modes (in fact, for MAC values exactly equal to 0 the modes are orthogonal).

Choice of parameters.

The selection of the parameters to be updated is a crucial step. Usually in iterative model updating the parameters are chosen to represent physical quantities, like the Young's modulus, the Poisson's coefficient, densities and geometrical properties. In the first place, these is done to retain good physical understanding of the finite element model.

Parameter selection heavily influences the posedness of the updating problem (Friswell & Penny, 1992). Generally, good practices to avoid ill-conditioning or ill-posedness are (1) chose physically meaningful updating parameters that adequately affect the model output and (2) reduce the number of parameters to limit the occurrence of under-determinacy issues in the updating problem (Ahmadian, Gladwell, & Ismail, 1997). The first task may be accomplished by making use of sensitivity-based methods to discard non-sensitive parameters, the second by dividing the structure in sub-parts with the same material properties. Additionally, the richness and the nature of the measured data, in contrast to the degree of discretization of the finite element model, places a limit to the type and number of parameters we can update to retain physical meaningfulness. A number of parameterization approaches has been developed by researchers. Substructure parametrization, heuristic subset selection (Lallement & Piranda, 1990) and parameter

clustering (Shahverdi, Mares, Wang, & Mottershead, 2009) are among the better-known ones.

In summary, choosing the right parameters is challenging and requires fine engineering judgement, since the success of the updating procedure requires keeping the number of parameters low enough to ensure uniqueness and selecting parameters for which the output is sensitive enough to avoid matrix singularities, while upholding the capability of the parameters to abate the measured/computed data misfit.

In sensitivity-based methods when the parametrization doesn't result in well-conditioned problems (for example, when a large set of candidate parameters is anyway required), use can be made of regularization techniques (Hansen, 1998), (Neumaier, 1998). Regularization consists in forcing well-conditioning by perturbing the cost function in a way that leads to an adjacent solution, but at the same time ensures the posedness of the problem.

Ill -conditioning and ill-posedness.

The posedness of the updating problem, as mentioned, is deeply influenced by the chosen updating parameters and the nature of the measured data. Moreover, various issues of ill-conditioning or rank-deficiency may arise in relation to the specific optimization technique used. For example, in the case of the Bayesian optimization approach that will be used in this work and explained in detail in Paragraph 2.3, the rank of the covariance matrix of the Gaussian Process (i.e., correlation matrix or kernel matrix) may be source of some concern. The matrix can become nearly singular if (1) the original function that is being reconstructed is so smooth and predictable that leads to a high-correlation between sampling points, thereby generating columns of near-one values, (2) the sampled points are very close one to another (which typically happens towards the end of the optimization process), thereby generating several columns that are almost identical.

To showcase posedness issues in case of an unwise choice of the updating parameters, a very simple 3 degrees of freedom shear-type frame with columns characterized by equal

stiffness is considered. The dynamic system is formally described by the following matrices:

$$M = \begin{bmatrix} m_1 & 0 & 0 \\ 0 & m_2 & 0 \\ 0 & 0 & m_3 \end{bmatrix} \qquad K = \begin{bmatrix} 2k & -k & 0 \\ -k & 2k & -k \\ 0 & -k & k \end{bmatrix},$$

Figure 1 (left) shows the cost function (defined as in Paragraph 4.1.1) plotted against the mass of the second floor m_2 and the stiffness k ranging from $5/4$ to $5 \cdot 2$ and from $10^5/4$ to $10^5 \cdot 2$ respectively, while constraining m_1 to be 20% less than m_2 and m_3 to be 20% higher than m_2 (meaning, for example, that we precisely know the volumes of the floors but are unaware of the material density). The cost function evaluates the misfit between the measured and computed output making use of the three natural frequencies and the MAC values associated to the three mode shapes of the system. As the cost function is minimized for infinite combinations of k and m_2, the solution to this updating problem is not unique.

Of course, the direct consequence of such ill-posed cases is that the results of several optimization runs are not consistent.

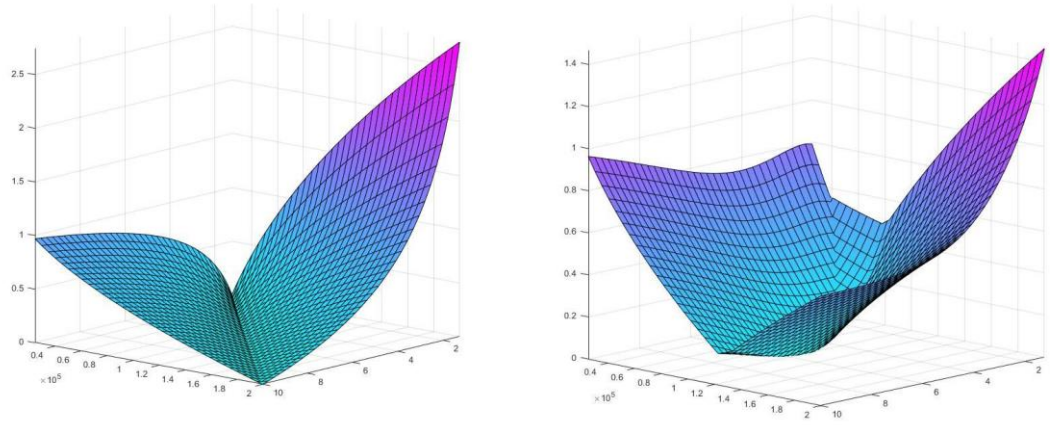


Figure 1 On the right, m_1 is kept 20% less than m_2 and m_3 20% higher than m_2; on the left, m_2 and m_3 are fixed. Uniqueness of solution is obtained in the second case only.

The cost function of a similar updating problem is likewise represented in Figure 1 (right): in this case, the values of m_1 and m_3 are fixed (meaning that we have no uncertainty

around these two quantities). The somewhat subtle change is enough to ensure the well-posedness of the problem as this time the optimization of the cost function leads to a unique solution.

In summary, choosing the parameters to be updated while retaining the well-posedness of the problem is non-trivial, especially in complex FE models. Moreover, one should be particularly careful when updating both mass-related and stiffness-related parameters, as chances are high of negatively affecting the posedness of the problem.

2.2 Traditional global optimization algorithms

When iterative optimization methods that make use of a penalty function are employed, model updating turns out to be a constrained optimization problem. As mentioned, the penalty functions of complex finite element models distinguish themselves for many dimensions, many local minima, high non-linearity, and non-smoothness. In this cases, traditional optimization algorithms might very well get stuck in local minima or fail to converge even in well-posed problems. In Section 4, the performances of the generalized pattern search (GPS) algorithm, the Genetic Algorithm (GA), the simulated annealing algorithm (SA) and the Bayesian optimization one are compared and benchmarked in several structural dynamics finite element model updating applications.

GPS is a relatively simple traditional optimization algorithm, while the other three are generally considered to be "computational intelligence" optimization techniques. All four algorithms have in common that no use of the derivatives is made, hence the function is not required to be differentiable. Despite the different approaches and backgrounds, Simulated annealing, GA and Bayesian optimization techniques share many elements: all algorithms are designed to carry out a global search of the minimum, avoiding local minima; they behave well for non-linear and non-smooth functions; the algorithms (except for GPS) show good robustness, since the set of initial parameters has little or no influence on the final results. The key difference between the Bayesian optimization approach (which exploits a surrogate model to predict the output of the objective function) and the other techniques is that the former requires a lower number of function evaluations, greatly enhancing the sampling efficiency: this really comes to a hand for expensive objective functions, which evaluation involves the computation of a complex finite element model. On the other hand, a greater sampling efficiency is the outcome of a more sophisticated algorithm, as the Bayesian optimization approach involves computationally intensive operations at each iteration to fit the surrogate model and choose the next sampling point. Finally, GA, simulated annealing and Bayesian optimization techniques tend to give results that are around the global minimum, but not extremely accurate: if more accuracy is needed, a deterministic and more traditional

algorithm that offers high accuracy when determining the local minimum, such as the GPS algorithm, can be initialized from the result (i.e., exploration through the probabilistic search optimization, and exploitation through the deterministic algorithm by allowing a few more iterations).

A brief explanation of the GPS algorithm, the GA and the simulated annealing optimization techniques is given in the following paragraphs, while the Bayesian optimization approach is discussed in more detail in Paragraph 2.3.

2.2.1 Generalized Pattern Search algorithm

The Generalized Pattern Search (GPS) algorithm is the simplest and less sophisticated algorithm among the optimization techniques compared in this work. It is a direct search non-probabilistic technique, that doesn't rely on the computation of the gradient of the objective function, the first formulation of which was given by Hooke and Jeeves in 1961 (Hooke & Jeeves, 1961). Despite its simplicity, it has been successfully applied for many scopes in the last decades. The key element of a GPS algorithm is the *pattern*, by which the algorithm individuates a set of points (called *mesh*) surrounding the current one where the objective function is evaluated at each step.

The mesh of points is formed by adding the current point to a scalar multiple of a set of vectors (the pattern). If one point of the mesh has a lower objective, it becomes the new incumbent point, and the algorithm moves accordingly. The number of vectors that constitutes the pattern is defined by the dimension of the problem and the positive basis set. Typically, the maximal basis (with $2d$ vectors) and the minimal basis (with $d + 1$ vectors) are used. The maximal basis is formed by the ortho-normal basis and the basis opposite in sign. When using the maximal basis, the algorithm performs $2d$ evaluations of the objective function at each iteration.

The phase during which the algorithm evaluates the objective in the mesh points is commonly called *polling*. Depending on the implementation, the algorithm may stop as

soon as it encounters a lower value of the objective in one of the points (such approach is adopted here), or it is required to execute a complete poll and then select the point showing the best function value. Either case, when a better point than the current one is found, the poll is called to be successful. On the contrary, when no better point than the incumbent is found, a poll is called unsuccessful.

In case of a successful poll, the algorithm "centers" itself on the new best point and the research mesh is *expanded*: the scalar multiple, called the mesh size, is incremented (typically, multiplied by 2) to foster the search of the minimum in other areas of the optimization domain. In case of an unsuccessful poll, the algorithm stays on the current point and the research mesh is *contracted*: the mesh size is reduced (typically by a factor of 0.5) to promote the exploitation of the current potential minimum.

The choices and the parameters that affect and control the behavior of the generalized pattern search algorithm are:

- The vector basis that describes the research mesh (the maximal basis is adopted in the algorithm implementation used further on).
- The initial mesh size.
- The mesh scaling parameters: the contraction factor and the expansion factor. These will be chosen equal to 0.5 and 2, respectively.

While the GPS technique is very simple, consequently leading to a very light algorithm, many evaluations of the objective function are performed, especially in high-dimensional problems. This is a major drawback of the algorithm when optimizing a computationally expensive function. Furthermore, this technique may involve a high risk of converging to a local minimum rather than to the global optimum.

2.2.2 Genetic Algorithms

A Genetic Algorithm is a probabilistic technique designed to find an approximate solution to difficult optimization problems. Initially proposed by (Holland, 1992), the algorithm

seeks the best solution among a population of solutions: this is achieved by the application of the principles of evolutionary biology to the optimization problem. More precisely, it moves from the hypothesis that individuals that have a genetic advantage over the others also have higher chances to breed successfully. During a cycle of generations, new individuals with an enhanced genetic makeup are produced through selection and recombination operations, in analogy to the way genes are biologically transferred between individuals. As these advantaged individuals spread their genes across the entire population, this gradually leads to an improved overall fitness.

The three most important phases in the implementation of a genetic algorithm are:

- the definition of the fitness function;
- the definition of the genetic representation of the individuals;
- the definition of the genetic operators (reproduction, crossover, and mutation).

The fitness function assigns a fitness value to each chromosome, therefore evaluating their aptness to solve the optimization problem. The fitness value of the chromosome governs its probability to reproduce, with particularly fit chromosomes having high chances to breed, and weak chromosomes having little or no probability to breed.

Regarding genetic representation, the basic idea is that the genetic makeup of an individual is represented by a chromosome, which is described by a binary-string. The number of chromosomes in the population equals the number of sampled points during one iteration, as each chromosome represents a vector of updating parameters. To univocally generate a chromosome from an input vector of parameters, a coding technique is needed. For example, a common coding method is the Gray encoding (Levin & Lieven, 1997): such method is able to code numbers into binary strings by changing only one bit. When using standard binary encoding, adjacent numbers may encode to completely different binary strings, therefore creating obstacles that are difficult to overcome by an algorithm that relies on genetic operators. The bit-depth of the encoding process gives the degree of resolution for each parameter range: for example, a 10-bit precision discretizes the rage of each parameter in 1024 equally spaced points.

To simulate the biological reproduction, the GAs use genetic operators. The three principal operators are the *reproduction operator*, the *crossover operator* and the *mutation operator*; these are applied to the extant population to generate a new population of chromosomes that is possibly more evolutionary-fit. The reproduction operator assigns a probability to be reproduced to each chromosome, based on their fitness. This probability may be determined directly by the fitness function value, or by mapping the fitness value to pre-defined probability batches (ranking). Hence, the ranking function controls the rate of convergence of the algorithm. The crossover operator, instead, handles the merging process of two chromosomes at the moment of reproduction. The two binary strings are cut at a random point, then, the first part of one chromosome is rejoined to the remaining part of the other chromosome to generate the new one. The crossover operator can potentially lead to fitter chromosomes by mixing successful chromosomes, but it may also result in the loss of the best chromosomes of a population. For this reason, some implementations make provision of an elite of chromosomes of fixed size N_e, constituted by the best chromosomes of the population, that doesn't experience any change during the reproduction process. A pair of chromosomes has a probability p_c of undergoing the crossover operation. The mutation operator has the purpose of randomly modifying the genetic information in the population, acting similarly to biological mutation. The probability p_m that mutation occurs in each chromosome is rather small, usually about 1%.

In summary, the GA workflow is as follows: a population of size N_p is initialized, where each chromosome represents an input vector of parameters (function sampling point) with a chosen resolution along the updating range; the objective function is sampled at each point determining the fitness values; each chromosome is assigned a probability to be reproduced by the reproduction operator according to its fitness value; each chromosome of the population undergoes the crossover operator and the mutation operator with a probability of p_c and p_m, respectively; a new population is created and the entire process is iterated until a convergence criterion is met.

The parameters that control the behavior of the GA and its effectiveness are:

- The population size N_p (many criteria can be followed to come up with a value for N_p. The rule of dumb used in this work is described in Section 3)
- The crossover probability p_c; usually in the range of 0.5-0.8.
- The mutation probability p_m; usually in the range of 0.01-0.001.
- N_e and N_b (N_b is the number of weakest chromosomes to be replaced with new ones at each iteration).

2.2.3 Simulated Annealing algorithms

Simulated annealing algorithms are used to find the global minimum in an optimization problem, which objective is characterized by many local minima. First used in optimization problems by (Kirkpatrick, Gelatt, & Vecchi, 1983), the concept of simulated annealing techniques comes from an analogy to the annealing treatment of physical materials (as metals). When annealing, a material is heated to the melting point and then very slowly cooled to the freezing one, so that the material is *approximately* always in thermodynamic equilibrium. Since many materials, as well known for metals, have multiple stable states, according to the temperature, that corresponds to states of minimum energy, if the temperature is lowered quickly enough, the material may be trapped in a metastable state at the freezing point, hence in a local energy minimum. This process is known as quenching. On the contrary, if temperature is lowered slowly enough (eventually infinitely slowly) the material is guaranteed to end up in the global minimum-energy state when freezing occurs.

(Metropolis, Rosenbluth, Rosenbluth, Teller, & Teller, 1953) proposed an algorithm to simulate the annealing process at the computational level. According to this technique, the input parameter values represent the state of the system, the objective (or fitness) function represents the energy function, and a parameter that controls the optimization (annealing) process is seen as the temperature. The simulation consists in randomly

perturbating the current state through a problem-dependent neighborhood function, then, the fitness of the new state is computed. If the fitness of the new state is higher than the old one, the new state is always accepted. If the fitness is lower, the new state is accepted with a certain probability (probability of acceptance), which depends on the fitness difference and the temperature. The neighborhood function, that controls the extent of the search, also depends on the control parameter (the temperature). As the temperature decreases according to an annealing schedule, the extent of the search as well as the probability of accepting a less fit state reduce, and the solution converges to an optimum.

As such, the algorithm works by performing the following steps:

- To initialize the procedure, an initial parameter input vector (initial system state) is randomly generated, and its fitness values is computed.
- A new trail point is chosen by the neighborhood function: this describes the distance of the new point from the current one by a probability distribution with a scale dependent on the current temperature.
- The fitness of the new point is computed: if the point is fitter, it is accepted and becomes the next point; if the new point is less fit, it is accepted with a probability given by the acceptance function. According to the Metropolis criterion, which probability of acceptance stems from the Boltzmann distribution, the acceptance function is:

$$\frac{1}{1+\exp\left(\frac{\Delta}{max(T)}\right)},$$

  where Δ is the difference between the fitness values and T is the temperature parameter. As the temperature decreases or delta increases, the probability of acceptance reduces.
- The temperature is lowered at each iteration according to a function that regulates the cooling schedule.
- Reannealing is eventually introduced, depending on the SA strategy adopted. Reannealing is the process of raising the temperature again, after a certain number of new points has been accepted, to enhance the aptitude of the algorithm to carry out a global search of the optimum and escape local minima.

- The algorithm stops when a certain convergence condition is met.

Depending on the approach used, many neighborhood functions may be employed. A way is to choose the new point with a step length equal to the current temperature value, in a direction that is uniformly random. This approach is adopted in the implementation used in this work.

Similarly, many cooling schedules can be adopted. The cooling rate should be sufficiently low to approach the condition of thermodynamic equilibrium in order to find the global energy optimum (Arnab & Chakrabarti, 2005). Of course, the drawback of a too low cooling rate is that the algorithm converges very slowly, requiring the objective function to be evaluated many times. Commonly adopted cooling schedules are $T = T_0/log(k)$, $T = T_0/k$ or $T = T_0 \cdot 0.95^k$, where k is the iteration number and T_0 is the initial temperature. (Ingber, 2000) provides additional information on the SA algorithm as well as an excellent review of the many shades SA can assume when following different implementation approaches. In the three case-studies analyzed in this work, two different SA strategies have been developed for the minimization of the objective function. For details about the implementation of both strategies, refer to Paragraph 3.2.

2.3 Bayesian optimization

When dealing with expensive functions to optimize, efficiency in terms of sampling is a fundamental requirement. Moreover, when functions are non-convex, an optimization algorithm must be able to discard local minima and search for the global optimum. These two requirements are somewhat conflicting, as the most straight forward approach we may think to enhance our probability of finding the global optimum is to increase the sampling volume, which is the very thing that we also want to avoid. Indeed, a number of global optimization techniques have been developed during the years, like the ones treated in the previous paragraph, but very few perform well when the number of function evaluations is kept at a minimum. One way to deal with the optimization of expensive functions, so when efficiency is paramount, is by using surrogate optimization techniques. This approach consists in substituting the objective function with a fast surrogate model (or response surface), which is then used to carry out the search of the optimum so to speed up the optimization process. Of course, the validity of the surrogate model, that is to say its capability to represent the behavior of the underlying objective function, is of uttermost importance to obtain good and reliable results. Generally, it is very difficult to find a functional form that can reconstruct the objective with reasonable accuracy, but this task becomes actually impossible when we have access to none or scanty a priori information about the function of interest, so in the case of black-box function optimization. In other words, when a linear regression of the form

$$y\left(\mathbf{x}^{(i)}\right) = \sum_{h} \beta_h f_h\left(\mathbf{x}^{(i)}\right) + \epsilon^{(i)} \qquad (i = 1, \dots, n)\,,$$

is used to fit the data (where $\mathbf{x}^{(i)}$ is the i-th sampled point out of a total of h, $y\left(\mathbf{x}^{(i)}\right)$ is the associated objective value, $f_h(\mathbf{x})$ is a function of $\mathbf{x}$, β_h are coefficients to be estimated, and $\epsilon^{(i)}$ are the independent errors, normally distributed), it is arduous to determine which functional form that fits well we should use, given we are dealing with a black-box model. As such, this kind of approach are impracticable in model updating optimization problems.

The approach used in Bayesian optimization consists of a change of paradigm for what concerns the surrogate model. Instead of trying to minimize the error $\epsilon^{(i)}$ by selecting some functional form that fits well the data, focus is placed on modeling the error by means of a stochastic process, so that the surrogate model is of the form:

$$y\left(\mathbf{x}^{(i)}\right) = \mu + \epsilon\left(\mathbf{x}^{(i)}\right) \qquad (i = 1, \dots, n)\,,$$

where μ is the regression term (the functional form is a constant), and the error term $\epsilon\left(\mathbf{x}^{(i)}\right)$ is a stochastic process with mean zero, so in other words a set of correlated random variables indexed by space. This change of perspective about the surrogate function is comprehensively described in one of most interesting papers on modern Bayesian optimization, (Jones, Schonlau, & Welch, Efficient Global Optimization of Expensive Black-Box Functions., 1998), where the proposed method is called Efficient Global Optimization, EGO. Besides modeling the surrogate as a stochastic process, the Bayesian optimization method makes use of an acquisition function to perform a utility-based selection of the points to be sampled. These are in fact the two key elements in Bayesian optimization. In the following paragraphs, after introducing the Bayesian approach moving from the concept of Bayesian inference, the probabilistic surrogate model and the acquisition functions will be illustrated, and the effects of different choices over these two key elements will be discussed.

Bayesian optimization has gained more attention only in the last decades, despite the first works on the topic are from Kushner (1964), who made use of Wiener processes and a search model formulated on the maximization of the probability of improvement. After some important developments by Mockus (1978), that also used Wiener processes, the concept of Bayesian optimization using Gaussian Processes (GP) as surrogate model was first used in the EGO formulation, that combines the DACE "Design and Analysis of Computer Experiments" model (Sacks, Welch, Welch, & Wynn, 1989) with the expected improvement concept.

In the last years plenty of research work and engineering applications have proven the benefits of using Bayesian optimization with expensive non-convex functions (Bobak

Shahriari, 2016), and as such it has become a popular and well-known global optimization technique.

As shown in the following paragraphs, Bayesian optimization consists in modeling the objective function by means of a probabilistic surrogate, and taking advantage of the probabilistic features of the surrogate model to wisely drive the objective function sampling by means of an acquisition function. In many cases, fitting a surrogate model to the observations requires to solve an optimization process to determine some hyperparameters. Similarly, the point to be sampled corresponds to the maximum of the acquisition function, which means that another optimization process is necessary at this step. Hence, the Bayesian optimization approach typically entails two secondary (relatively cheap) optimization problems: this results in a somewhat fancy and potentially heavy algorithm, which use makes sense only if the objective function to be optimized is reasonably expensive to compute.

To present a brief description of the theory behind the Bayesian approach, the following notation will often be used:

$$\mathcal{D}_{1:t} = \{\mathbf{x}_{1:t}, f(\mathbf{x}_{1:t})\}\,.$$

This is the observations set, or sample, made of t observations in total. $\mathbf{x}_i$ is the input point vector of the i-th observation. This vector, in other words, contains the updating parameters (in the input space). The length of $\mathbf{x}_i$ equals d, the number of dimensions of the updating problem, i.e., the number of updating parameters. Finally, $f(\mathbf{x}_{1:t})$, sometimes abbreviated in $\mathbf{f}_t$, are the values of the objective function at $\mathbf{x}_{1:t}$, i.e., the outputs of the penalty function for each set of updating parameters $\mathbf{x}_i$.

2.3.1 Bayesian inference in optimization problems

As already mentioned in Paragraph 2, Bayesian optimization derives its name by the application of the Bayes' theorem to the optimization problem. In fact, the prior, as found in the theorem, is a stochastic process that represents our belief about the behavior of the objective function. In particular, we may have some beliefs about the degree of smoothness of the function, and this is addressed by the choice we can make on the stochastic process we use (for Gaussian Processes, the kernel functions and the related hyperparameters govern the degree of believed smoothness, as it will be shown in the following paragraph).

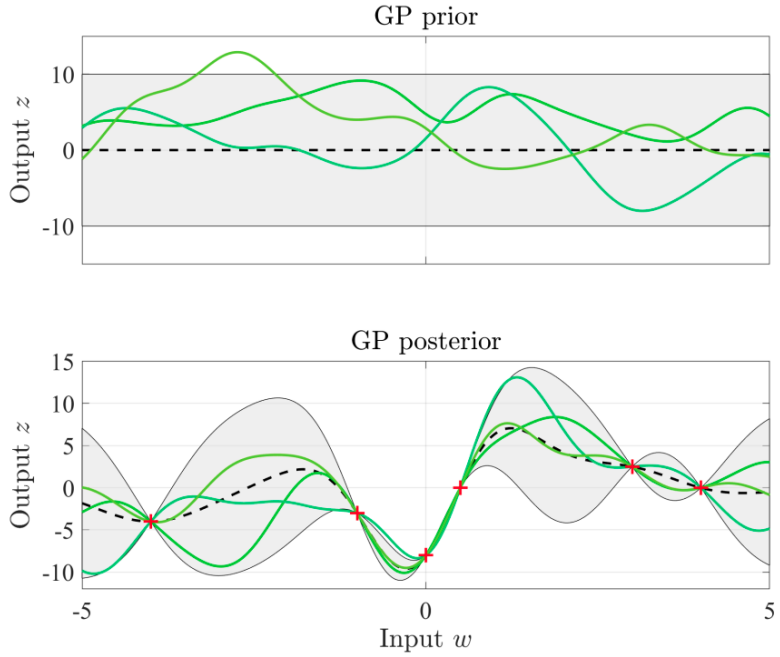


Figure 2. At the top, a GP prior, that represents the initial belief about the objective function. At the bottom, a GP posterior, that represents the updated belief about the objective function, given the new observations (red points). The dashed line represents the mean of the GP (i.e., the predicted objective function values), the gray bands are the variance about the predicted values, and the green lines represent function samples from the GP. *(Maiworm, Limon, & Findeisen, 2021)*.

The *prior* probability $P(f \mid \mathcal{D}_{1:t-1})$, that is represented, in broad sense, by a stochastic process at a stage $t-1$ of the optimization procedure, is updated by adding a new observation, that consists in sampling the function at a certain point, yielding the posterior

(or updated) probability $P(f \mid \mathcal{D}_{1:t})$, that is represented by the updated stochastic process a the stage t of the optimization procedure. As already discussed, this can be practically seen as a sort of inference reasoning (see Figure 2).

So, the posterior depicts our updated beliefs about the underlying objective function. Nonetheless, the stochastic process that is used to model the posterior probability actually acts as a surrogate (probabilistic) model of the unknown objective. As such, Bayesian optimization can be interpreted as a surrogate optimization technique, which is the interpretation already given about this optimization method.

2.3.2 Probabilistic surrogate model: Gaussian Process priors

In theory, any probabilistic (stochastic) model can be adopted to describe the prior and the posterior, and so as a surrogate of the underlying objective function. The model must be probabilistic, since any output relative to any $\boldsymbol{x}$ input point describes a probability distribution and, basically, must provide a prediction and an uncertainty about that prediction.

Practically, the probabilistic model should satisfy some requisites. First, the model should be relatively light and fast, meaning that the computation of the first and second central moments (expected value and variance) is an easy and non-expensive operation. In fact, in a way we are using the surrogate model not only as an aid to "intelligently" select the points to be sampled (by taking advantage of its probabilistic features by means of the acquisition functions) but also to speed up the optimization process. From this perspective, the use of a computationally expensive predictive model (potentially even more so than the objective function itself) makes not much sense. Second, the probabilistic model employed should be able to adequately fit the objective function with a small number of observations if optimization efficiency in terms of sampling is pursued. Third, one of the conditions to ensure the convergence of the BO method is that the conditional variance must cancel if and only if the distance between an observation and the prediction point is zero (Mockus J. , 1982).

Following these requisites, it comes as no surprise that Gaussian Process (GP) priors are the chosen probabilistic model in the majority of modern Bayesian optimization implementations. To mention some popular alternatives to GPs, (Hutter, Hoos, & Leyton-Brown, 2011) worked with random forests, (Snoek, et al., 2015) with deep neural networks, (Springenberg, Klein, Falkner, & Hutter, 2016) made use of Bayesian neural networks, while (Wang, Gehring, Kohli, & Jegelka, 2018) used Mondrian trees. GP are well-suited for model updating problems where the penalty (black-box) function to be minimized is continuous.

A Gaussian Process is a collection of random variables, any finite number of which have consistent joint Gaussian distributions (Rasmussen C. , 2004), that is completely defined by a mean function and a covariance function over $\boldsymbol{x}$, just like a Gaussian distribution is completely specified by its mean and covariance. According to this definition, we can describe a GP with

$$f(\mathbf{x}) \sim \mathcal{GP}\big(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')\big)\,,$$

where $m(\mathbf{x})$ is the mean function (that is the prediction about the objective value), and $k(\mathbf{x}, \mathbf{x}')$ is the covariance function (that is the uncertainty about the prediction). Intuitively, a GP can be imagined as a function that returns, for any input point $\mathbf{x}$, a gaussian distribution over the possible values of the objective $f(\mathbf{x})$, described by its mean and variance (Figure 3).

Without loss of generality, the mean of the GP $m(\mathbf{x})$ can be considered equal to zero, and focus can be placed on the covariance function $k(\mathbf{x}, \mathbf{x}')$. The correlation value k is derived as follows. The basic idea is to consider the level of correlation between two observations, f_i and f_j, as related to the distance between the points corresponding to those observations in the input space, x_i and x_j. The reasoning behind this concept is that, intuitively, we expect a high level of correlation on the output of two points very close to each other, and, on the contrary, a low correlation (or no correlation at all) if the two points are very far from each other. Consequently, a decreasing function of the distance between the points in the input space can be chosen. Many covariance functions $k(\mathbf{x}, \mathbf{x}')$ (also known as *kernel functions*) may be selected (in the following paragraphs, four

popular kernel functions are discussed). One of the most employed, the "squared exponential" function, is of the form:

$$k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{1}{2}\|\mathbf{x}_i - \mathbf{x}_j\|^2\right).$$

Coherently with what stated, this function equals 0 for an infinite distance $\|\mathbf{x}_i - \mathbf{x}_j\|$, and equals 1 when the distance is zero.

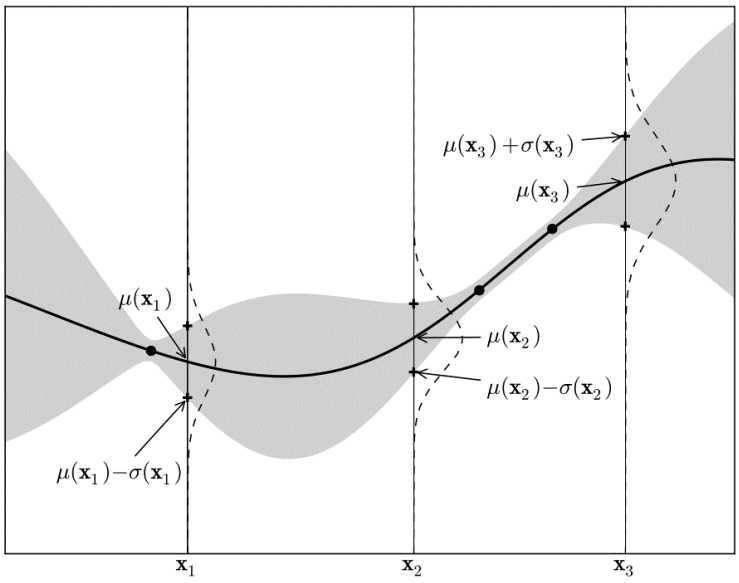


Figure 3. A representation of a Gaussian Process: the dots are the objective observations, the black line is the GP prediction, while the gray bands represent the variance about the prediction. At input points x_1, x_2 and x_3, it is possible to see the gaussian distribution completely described by the mean and the variance of the GP.

Considering the set of t observations $\mathcal{D}_{1:t} = \{\mathbf{x}_{1:t}, f(\mathbf{x}_{1:t})\}$, the covariance can be computed for each pair of sampled points and conveniently arranged in matrix form as follows:

$$\mathbf{K} = \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & \dots & k(\mathbf{x}_1, \mathbf{x}_t) \\ \vdots & \ddots & \vdots \\ k(\mathbf{x}_t, \mathbf{x}_1) & \dots & k(\mathbf{x}_t, \mathbf{x}_t) \end{bmatrix}.$$

This is called the covariance matrix, or *kernel matrix*, and it really stands at the core of the Gaussian Process, as the name suggests. The diagonal terms are obviously equal to 1, as the output at any point is perfectly correlated with itself, in case of noiseless

(deterministic) objective functions, which is the case of model updating when no uncertainty over the measured output is considered and a fully deterministic FEA is employed (which will be the case of the optimization environment of model updating problems considered in this work).

Since the GP is used for an utility-based selection of the next point to be selected, which is done by means of the acquisition function, the expected value and the related variance must be computed at any chosen point $\boldsymbol{x}_*$. To do that, we can consider the joint Gaussian distribution:

$$\begin{bmatrix} \mathbf{f}_{1:t} \\ f_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{K} & \mathbf{k} \\ \mathbf{k}^T & k(\mathbf{x}_*, \mathbf{x}_*) \end{bmatrix}\right),$$

where f_* is the objective output at $\boldsymbol{x}_*$, that is $f_* = f(\mathbf{x}_{1:t})$, and

$$\mathbf{k} = [k(\mathbf{x}_*, \mathbf{x}_1) \quad k(\mathbf{x}_*, \mathbf{x}_2) \quad \cdots \quad k(\mathbf{x}_*, \mathbf{x}_t)]\,.$$

From this, the following *predictive distribution* can be derived (for a full analytical derivation, see (Rasmussen & Williams, 2006)):

$$P(f_* \mid \mathcal{D}_{1:t}, \mathbf{x}_*) = \mathcal{N}\big(\mu_t(\mathbf{x}_*), \sigma_t^2(\mathbf{x}_*)\big)\,,$$

where

$$\mu_t(\mathbf{x}_*) = \mathbf{k}^T\mathbf{K}^{-1}\mathbf{f}_{1:t} \tag{1}$$

$$\sigma_t^2(\mathbf{x}_*) = k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}^T\mathbf{K}^{-1}\mathbf{k}\,. \tag{2}$$

In the above set of equations, $\mu_t(\mathbf{x}_*)$ is the prediction over the objective function value (the mean of the GP) at any chosen point $\mathbf{x}_*$, and is $\sigma_t^2(\mathbf{x}_*)$ the variance of the prediction at $\mathbf{x}_*$ (the subscripts here denotes that the perdition and its variance come from a GP trained over the $\mathcal{D}_{1:t} = \{\mathbf{x}_{1:t}, f(\mathbf{x}_{1:t})\}$ sample data).

From the (1) and (2) (by means of which exact inference is computed), it is clear that in order to compute a prediction and the related variance from a GP, it is necessary to invert the kernel matrix $\mathbf{K}$. This operation has a computational complexity of $\mathcal{O}(N^3)$, where N is the size of the (square) kernel matrix (that equals the number of observations, t). While this operation is relatively cheap on its own, it can in fact lead to computationally

burdensome workflows as (1) the Bayesian optimization approach entails the maximization of the acquisition function (as discussed in Paragraph 2.3), a task that may require computing thousands of predictions, especially in high dimensional problems, and (2) the number of observations keeps increasing (and so the size of **K)** as the optimization advances (a new point is sampled at each iteration). Therefore, when using Gaussian Processes, the optimization procedure badly scales as the number of observations grows. One way to mitigate such a problem consists in limiting the number of observations used to fit the GP to a certain amount (e.g., defining an "active set" size of a few hundreds), by randomly choosing the fitting points among the sample at each iteration of the algorithm. This practice is applied in the implementation used within this work.

2.3.2.1 Kernel functions in Gaussian Processes

The covariances that appear in the kernel matrix are determined by the kernel functions, which deeply affect the smoothness properties of the GP. Of course, these must be coherent with the properties of the underlying objective function in order to get good predictions. As each problem has its own specifics, the kernel function, that determines the degree of correlation on the response based on the distance between two points in the input space, such as the one seen in the previous paragraph, must be properly scaled. To achieve this, the kernel functions are generalized by introducing *hyperparameters*. In case of a squared exponential function, these results in the equation:

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma_f \exp\left(-\frac{1}{2\theta^2}\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2\right),$$

where σ_f is the *vertical scale*, which is the process standard deviation (i.e., describes the vertical scaling of the GP variance, as visible in Figure 3), and the hyperparameter θ is the *characteristic length scale*, which defines how far apart the input points $\boldsymbol{x}_i$ can be for the output to become uncorrelated. Figure 4 shows how the length scale affects the kernel function and therefore the correlation: high values of θ correspond to higher correlation at a given distance, low values of θ lead to lower correlation.

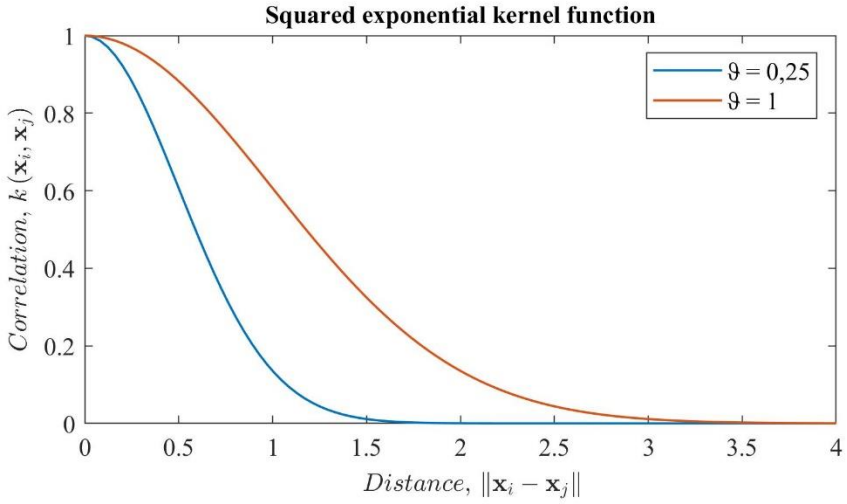


Figure 4. The squared exponential function is plotted with a scale equal to 0.25 (in blue) and to 1 (in red). Given the same distance, a higher length scale means higher correlation.

For isotropic models, one hyperparameter is sufficient as the problem has similar sensitivity to each parameter, that is to say changes in any of the parameters affect the system response to a similar extent. When dealing with anisotropic problems (as it is often the case with model updating problems), it is much more convenient to use separate length scales, one for each parameter. This is typically done through the use of automatic relevance determination hyperparameters (ARD), that consists in using a vector of hyperparameters $\boldsymbol{\theta}$, which size is equal to the number of updating parameters. In practice, when a certain length scale θ_l is high in value compared to the other length scales, the kernel matrix becomes independent on the l-th parameter, effectively discarding the dimension from the optimization procedure.

A procedure often followed to determine the optimal set of hyperparameters $\boldsymbol{\theta}$ is to compute and maximize the marginal log-likelihood of the evidence $\mathcal{D}_{1:t} = \{\mathbf{x}_{1:t}, f(\mathbf{x}_{1:t})\}$ given $\boldsymbol{\theta}$:

$$log\ (p(\boldsymbol{f}_{1:t} \mid \boldsymbol{x}_{1:t}, \boldsymbol{\theta}^{+})) = -\frac{1}{2}\boldsymbol{f}_{1:t}{}^{\top}\boldsymbol{K}^{-1}\boldsymbol{f}_{1:t} - \frac{1}{2}log\ |\boldsymbol{K}| - \frac{t}{2}log\ (2\pi)\,,$$

where the $\boldsymbol{\theta}^{+}$ vector contains the length scales $\boldsymbol{\theta}_{1:l}$, but also the vertical scale and the mean μ_0 (i.e., the constant regression term) of the GP (and therefore all the $d + 2$

hyperparameters), so that $\boldsymbol{\theta}^{+} := (\boldsymbol{\theta}_{1:l}, \mu_0, \sigma_f)$. In the previous equation, the dependency on $\boldsymbol{\theta}^{+}$ is obviously found in the kernel matrix $\boldsymbol{K}$.

When using ARD kernels, as the optimal set of hyperparameters is estimated by maximizing the likelihood, a sort of "sensitivity analysis" of the parameters over the sampled points is performed. This built-in feature of the Bayesian optimization technique may happen to be very useful, and it certainly is for what concerns structural model updating problems, where the system sensitivity to the updating parameters is often dissimilar and usually unknown.

Among the many possibilities, a good kernel for a specific problem is one that generates a valid surrogate model. A decision aid is given by cross-validation, which can be employed for the challenging task of selecting the right surrogate model to use, as clarified in Paragraph 2.3.2.2.

The definition of four popular kernel functions (unsquared exponential kernel, squared exponential kernel, and two kernels belonging to the Matérn function family, all of which are used in the case-studies discussed in Section 4), as well as the effects on the rendition of the underlying objective given by the surrogate model, follows hereafter.

- **ARD unsquared exponential kernel.**

  This kernel function is defined as:

  $$k(\boldsymbol{x}_i, \boldsymbol{x}_j \mid \boldsymbol{\theta}^{+}) = \sigma_f^2 \, exp\,(-D)\,,$$

  where $D = \sqrt{\sum_{l=1}^{d} \frac{(x_{i,l} - x_{j,l})^2}{\theta_l^2}}$.

  Among the four kernel functions considered, this generates the coarsest response surfaces.

- **ARD Squared exponential kernel.**

  This kernel is defined as:

  $$k(\boldsymbol{x}_i, \boldsymbol{x}_j \mid \boldsymbol{\theta}^{+}) = \sigma_f^2 \, ex\,p\left[\sum_{l=1}^{d} \frac{(x_{i,l} - x_{j,l})^2}{\theta_l^2}\right].$$

Among the kernel functions considered, this generates the smoothest response surfaces.

Another convenient choice is represented by the Matérn kernel functions (Matérn, 1960). These functions are defined as

$$k(\mathbf{x}_i, \mathbf{x}_j) = \frac{1}{2^{\varsigma-1}\Gamma(\varsigma)} \left(2\sqrt{\varsigma}\|\mathbf{x}_i - \mathbf{x}_j\|\right)^{\varsigma} H_{\varsigma}\left(2\sqrt{\varsigma}\|\mathbf{x}_i - \mathbf{x}_j\|\right),$$

where ς is a smoothness coefficient, while $\Gamma(\cdot)$ and $H_{\varsigma}(\cdot)$ are the Gamma function and the Bessel function of order ς, respectively. As the smoothness coefficient ς tends towards infinite, the Matérn function reduces to the squared exponential function; when ς tends towards zero, the Matérn function reduces to the unsquared exponential function. Two popular ARD Matérn kernels are:

- **ARD Matérn 3/2.**

  This kernel is defined as:

  $$k(\boldsymbol{x}_i, \boldsymbol{x}_j \mid \boldsymbol{\theta}^+) = \sigma_f^2 \, (1 + \sqrt{3}\, D) \, exp \, (-\sqrt{3}\, D),$$

  where $D = \sqrt{\sum_{l=1}^{d} \frac{(x_{i,l} - x_{j,l})^2}{\theta_l^2}}$.

  In this case, $\varsigma = \frac{3}{2}$. Hence, the smoothness properties of the response surface are closer to what obtained with the unsquared kernel function.

- **ARD Matérn 5/2.**

  This kernel is defined as:

  $$k(\boldsymbol{x}_i, \boldsymbol{x}_j \mid \boldsymbol{\theta}^+) = \sigma_f^2 \, (1 + \sqrt{5}\, D \, + \frac{5}{3}\, D^2) \, exp \, (-\sqrt{5}\, D),$$

  where $D = \sqrt{\sum_{l=1}^{d} \frac{(x_{i,l} - x_{j,l})^2}{\theta_l^2}}$.

  In this case, $\varsigma = \frac{5}{2}$. Hence, the smoothness properties of the response surface are closer to what obtained with the squared kernel function.

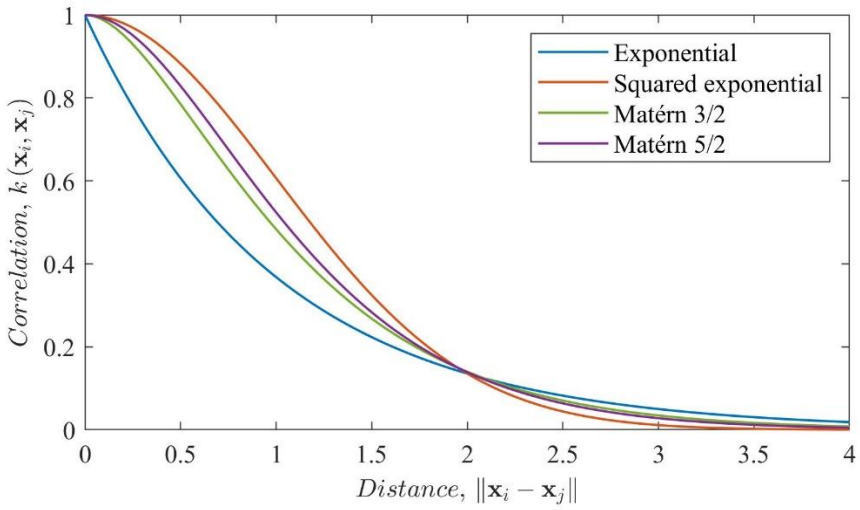


Figure 5. The four kernel functions are plotted against the distance. Correlation decreases quickly when using the unsquared exponential kernel, while the squared exponential kernel returns higher correlation for smaller distances. The Matérn functions show an intermediate behavior. Here, $\theta = 0{,}25$ is used.

Figure 5 enables to visualize the difference between the four kernel functions introduced above. The Exponential kernel, compared to the other kernels, is distinguished by a sharp drop in correlation as the distance increases.

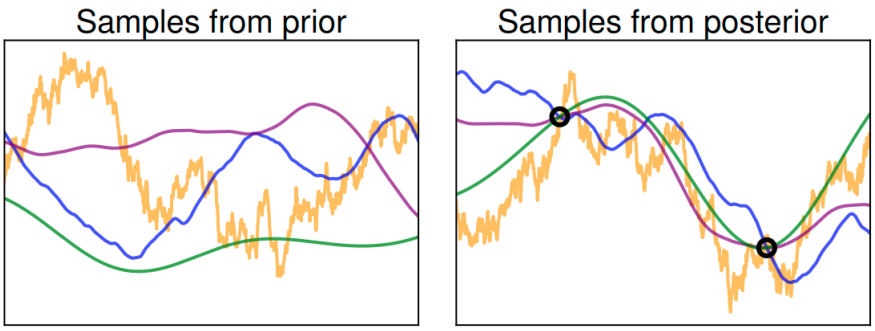


Figure 6. Function samples from four Gaussian Processes, built using the unsquared exponential kernel (yellow), the Matérn 3/2 kernel (blue), the Matérn 5/2 kernel (purple) and the squared exponential kernel (green). The sample obtained through the exponential kernel presents rougher features, while the sample generated by the squared exponential one shows smoother features. Overall, kernels are seen to have a huge impact on the GP rendition of the underlying objective function. (Shahriari, Swersky, Wang, Adams, & de Freitas, 2016).

The impact on function samples drawn from the different GPs built with the four kernels is instead visible in Figure 6. It is noticeable how the use of the exponential kernel

function, due to the sharper drop in response correlation, creates very rough features of the function sample, while Matérn 3/2, Matérn 5/2 and the squared exponential kernel functions generate increasingly smooth samples.

2.3.2.2 Surrogate model validation

Perhaps the most straightforward way to choose what kernel function should be used in Bayesian optimization procedure is to select one that generates the most valid surrogate model. In order to have an efficient optimization, a GP that generates samples which features are as close as possible to the objective function is needed, so that it can serve as a valid surrogate model, able to give good and quality predictions. *Cross-validation* is a powerful tool for checking the model validity, and therefore for establishing what kernel function is most suited to a specific problem.

Indeed, the surrogate model validity should be assessed not only to determine which kernel function to use, but also for other important scopes. The first, broader scope, is that we must be sure the probabilistic surrogate we are using is actually able to model the underlying objective function (Jones, Schonlau, & Welch, Efficient Global Optimization of Expensive Black-Box Functions., 1998). In fact, Gaussian Processes may not be ideal for some applications, where other kinds of stochastic models could provide much better results. Secondly, we may want to see if a transformation of the input variables (updating parameters) has an impact on the surrogate validity. For example, we may find that log transforming the input variables can lead to a better cross-validation of the GP used to model the objective function. This could be particularly true for model updating problems, as oftentimes a significant change of the input parameters has little impact on the modal properties of the system and therefore on the penalty function that measures the response misfit.

Cross-validation, also known as out-of-sample testing, is a technique that enable to assess how the results of a probabilistic analysis generalize to an independent dataset. The goal of cross-validation is to check the capability of a probabilistic model to predict new data that was not used to train the model itself. To this extent, one strategy consists in training

the model on a subset of the available observed data (the *training set*), and validating the model against the complementary subset (the *testing set*), that collects the left-out observations, by computing the error between the predictions and the left-out data. This procedure is known as data partitioning. Based on the type of partitioning, a cross-validation technique can be considered exhaustive or non-exhaustive. Exhaustive cross-validation returns exact and reproducible results, while non-exhaustive methods give approximated results, because not all ways of splitting the data are considered.

One popular exhaustive method is the *leave-one-out cross-validation* (LOOCV). The training set consists of only one observation, while the training set consists of all the remaining observations. This partitioning is repeated for all observations, which practically involves the training and the validation of n different models, where n is the total number of observations. When dealing with large datasets and/or the training of the model is expensive, LOOCV may require a large computational time and become unfeasible.

When this is the case, non-exhaustive approaches may be more appropriate. One common method that falls in this category is the *k-fold cross-validation*. The initial sample is partitioned in k equally sized subsamples: one subsampled is used as testing set, while the other $k-1$ subsamples are used to train the model. As a total of $\frac{n}{k}$ errors are computed, usually for each fold the mean squared error is considered (MSE). The cross-validation is then repeated a total of k times, so until each of the k subsamples has been used as validation data. Therefore, *k-fold cross-validation* involves the training of only k models, and the computation of n predictions in total, which comes at the benefit of computational time when working with large datasets of expensive predictive models.

Typically, the n errors computed by the LOOCV or the k MSEs found through a k-fold cross-validation are averaged together, to compute a final MSE value known as cross-validation loss, which is briefly indicative of the overall validity of the predictive model, based on the dataset at our disposal.

A Bayesian optimization procedure is usually initialized by seeding, that consists in sampling the objective function at a number of randomly chosen points. The size of the

initial seed can vary, but a general rule of dumb is to consider a number of points equal to $10\,d$ (Jones, Schonlau, & Welch, 1998), where d is, as usual, the number of dimensions of the optimization problem (i.e., the number of updating parameters). The seed points are used to fit a GP, which, other than serving as surrogate model for the Bayesian optimization procedure, can be typically used to perform cross-validation operations in order to (1) assess its overall validity, (2) select the most appropriate kernel function for the specific problem (by changing the kernel used in the GP) and (3) to determine whether a (log) transformation of the updating parameters provides some benefits in terms of predictive capability of the surrogate model.

The predictive model used in Bayesian optimization is probabilistic (a Gaussian Process in this implementation) which means that it is able to model the objective taking into account the uncertainty over the predictions made. As a consequence, the model should be evaluated not only against the predictions, but also taking into account the confidence that the model has about the prediction made. In other words, if a prediction is quite off from the test data, but also the uncertainty related to that prediction is high, we can say that the probabilistic model still validates well. To achieve this, typically the *standardized cross-validated residual* is conveniently considered (Jones, Schonlau, & Welch, 1998):

$$\frac{\mu_t(\mathbf{x}_{Test}) - f_{Test}}{\sigma_t(\mathbf{x}_{Test})},$$

where, accordingly to Eqns. (1) and (2), $\mu_t(\mathbf{x}_{Test})$ and $\sigma_t(\mathbf{x}_{Test})$ are the predicted function value and the related standard deviation at the testing point $\mathbf{x}_{Test}$ (here, the subscript t states that the model is trained on the $\mathcal{D}_{1:t} = \{\mathbf{x}_{1:t}, f(\mathbf{x}_{1:t})\}$ dataset), and f_{Test} is the test observation used for cross-validation. If the predictive model validates well, the standardized cross-validated residual values should roughly be in the interval $[-3, +3]$, as the model is 99.7% confident that the actual function value lies in the range $[\mu_t(\mathbf{x}_{Test}) - 3\,\sigma_t(\mathbf{x}_{Test}), \mu_t(\mathbf{x}_{Test}) + 3\,\sigma_t(\mathbf{x}_{Test})]$.

In summary, if the cross-validation of the Gaussian Process used a surrogate doesn't give positive results, some of basic strategies that can be adopted to improve the model validation may be:

- Logarithmic transformation of the input variables.
- Change of the kernel function used.
- Superposition of different kernel functions.
- Revising the set of kernel hyperparameters.

A good surrogate model validation lays the basis for an efficient, robust and fast converging optimization procedure.

2.3.3 Acquisition functions for Bayesian optimization

Along with the probabilistic surrogate model, the acquisition function is the second key element of the Bayesian optimization approach. The acquisition function has the role of guiding the search for the global optimum, by selecting the new point to be sampled taking advantage of the probabilistic features of the predictive model.

When considering the probabilistic surrogate model (e.g., a GP), two different approaches may be followed to pursue the optimum: the exploitative approach, which consists in sampling from areas where the objective function is expected to be low (i.e., where predictions of the surrogate model assume low values); and the explorative approach, which consists in sampling from areas characterized by high uncertainty (i.e., where the variance about the predictions of the surrogate model is high). The automatic tradeoff between exploitation and exploration is taken care of by the acquisition function.

Usually, in Bayesian optimization, the optimization problem is a maximization task, meaning that the objective function is searched for the maximum rather than the minimum. Hence, acquisition functions are defined so that high values of acquisition correspond to regions where the objective is potentially high in value. Therefore, when seeking the minimum of a function $f(\boldsymbol{x})$, such in the case of model updating, it is sufficient to consider the equivalent problem:

$$arg\,max_{\boldsymbol{x}}\, g(\boldsymbol{x}) \qquad g(\boldsymbol{x}) = -f(\boldsymbol{x})\,.$$

The next point $\boldsymbol{x}_{t+1}$ that will be chosen for sampling is found by maximizing the acquisition function $a(\boldsymbol{x})$ according to the optimization problem:

$$\boldsymbol{x}_{t+1} = arg\, max_{x}\, a(\boldsymbol{x}|\mathcal{D}_{1:t})\,.$$

Note that the objective value at $\boldsymbol{x}_{t+1}$, $f(\boldsymbol{x}_{t+1})$, is not guaranteed to be better than the incumbent value $f(\boldsymbol{x}^{+})$, where $\mathbf{x}^{+} = \text{argmax}_{\mathbf{x}_i \in \mathbf{x}_{1:t}}\, f(\mathbf{x}_i)$. Nonetheless, the added observation leads to an increased knowledge of the objective, which will be reflected by the updated Gaussian Process (the posterior) at stage $t+1$ of the optimization procedure.

Naturally, there are many acquisition functions that can be used to select sampling points, determining the global search attitudes, the convergence rate, and the overall sampling efficiency of the optimization procedure. In the following paragraphs, the definition of four common acquisition functions is given, showcasing how different functions make different choices when selecting the next sampling point.

2.3.3.1 Probability of improvement

Probability of improvement (PI) is one of the first acquisition functions used in the Bayesian optimization framework (Kushner, 1964). The definition of this acquisition function in its most rudimental form is given by:

$$\begin{aligned}\text{PI}\,(\mathbf{x}) &= P\big(f(\mathbf{x}) \geq f(\mathbf{x}^{+})\big) \\ &= \Phi\left(\frac{\mu(\mathbf{x}) - f(\mathbf{x}^{+})}{\sigma(\mathbf{x})}\right),\end{aligned}$$

where $\Phi(\cdot)$ is the CDF of the standard normal distribution. Looking at this definition, it is clear how the PI function actually penalizes points which prediction comes with high uncertainty (in fact, the standard deviation of the prediction is at the denominator). Such penalized points, from an explorative point of view, could instead lead to improved values of f. The approach of plain PI is therefore purely exploitative because it will prefer points with low uncertainty, even if these are associated with very low improvement values $\mu(\mathbf{x}) - f(\mathbf{x}^{+})$. When using PI, being this function extremely "greedy", the optimization procedure may converge very rapidly, but at the high risk of getting trapped in a local

minimum, rather than finding the global optimum of the objective function. Especially when the number of sampled points is low (relatively to the dimensions of the problem), this may lead to the wrong solution of the optimization problem.

A way to overcome these issues, and improve the explorative attitudes of PI, is to introduce a (strictly positive) trade-off parameter ξ:

$$\begin{aligned} \mathrm{PI}(\mathbf{x}) &= P(f(\mathbf{x}) \geq f(\mathbf{x}^+) + \xi) \\ &= \Phi\left(\frac{\mu(\mathbf{x}) - f(\mathbf{x}^+) - \xi}{\sigma(\mathbf{x})}\right). \end{aligned}$$

While this approach may look particularly appealing, since we can control exploration through the parameter ξ, it also raises some questioning on how to determine ξ so to ensure the right level of exploration, but at the same time promoting reasonable convergence rate throughout the optimization process (Jones, Schonlau, & Welch, 1998).

Figure 7 displays the PI function at work. Here, a simple numerical case consisting in a 3-DOF shear type system is considered. The columns are all identified by the same stiffens k, while the masses at each floor are m_1, m_2 and m_3. Two parameters are beaing updated: the stiffness k and one of the masses, m_2. The other two masses and the geometrical features are supposed to be known, and the respective values are therefore fixed. The penalty function (which, in this case, takes into account the three modes of the dynamic system) resulting from this updating setup is sampled at 9 randomly taken points: a Gaussian Process (the probabilistic surrogate model) is fitted to the observations and the acquisition function PI is computed by knowing $\mu(\mathbf{x})$ and $\sigma(\mathbf{x})$ (the prediction and the uncertainty given by the GP at $\mathbf{x}$, that can be cheaply computed). Since only two parameters are being updated, it is possible to visualize the mean of the GP (i.e., the predicted value of the objective function according to our surrogate) and the resulting acquisition function over the optimization space. In figure, the computed minimum point according to the surrogate model is visible, as well as the next point selected by the acquisition function to be sampled at the next iteration, which corresponds to the acquisition maximum.

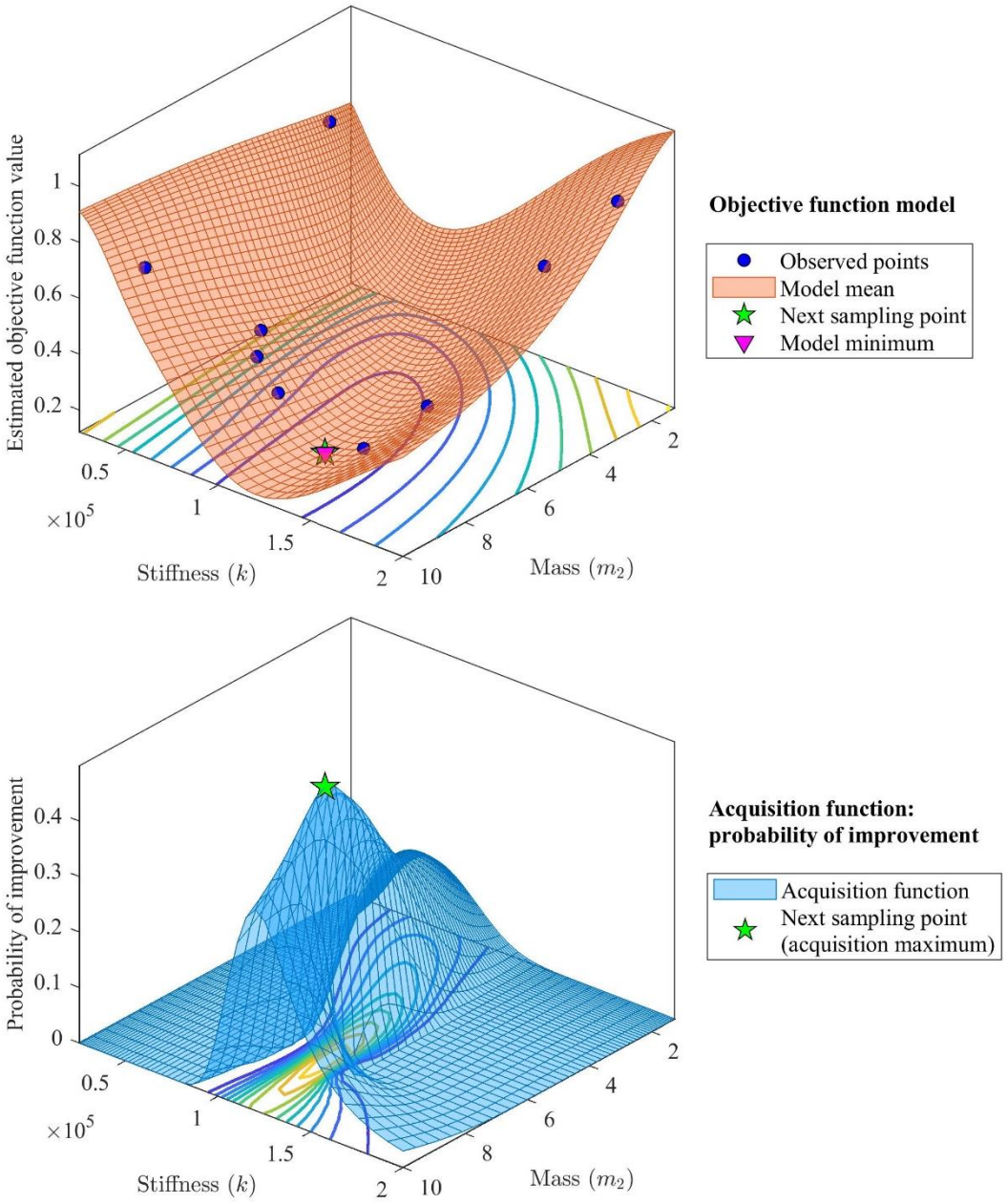


Figure 7. At the top: the plot represents the mean of the GP, $\mu_{1:9}(\boldsymbol{x})$, the nine observations $\boldsymbol{f}_{1:9}$, the model minimum (i.e., the lowest function value as predicted by the GP), and the next sampling point selected by the acquisition function probability of improvement (PI). At the bottom: the plot displays the acquisition function $PI(\boldsymbol{x})$ and the next chosen sampling point, which corresponds to the acquisition function maximum. Notice how the maximum of PI is found very close to the predictive model minimum, denoting its very aggressive sampling behavior.

As follows from its definition, PI shows a very "greedy" behavior, favoring a point which predicted objective value is (1) very low and close to the predictive model

minimum and (2) characterized by very low uncertainty. Of course, this results in a very fast converging optimization algorithm, but with a high risk of getting stuck in a local minimum. As a consequence, PI works best when the size of the initial seed relative to the dimensions of the problem (i.e., the number of initial observations) is large, since in this case the fitted surrogate model has greater accuracy and is therefore more trustworthy. When the objective function is suspected to be highly non-convex and the number of initial function evaluations must be kept small for computational reasons, PI is most probably not the best acquisition function to choose.

2.3.3.2 Expected improvement

Expected improvement (EI) concept is first found in the early work of (Mockus, Tiesis, & Zilinskas, 1978). This concept has the peculiarity of trying to find the point that corresponds the best expected improvement over the incumbent $f(\mathbf{x}^+)$, taking automatically into account both the improvement $\mu(\mathbf{x}) - f(\mathbf{x}^+)$ and the related uncertainty $\sigma(\mathbf{x})$ given by the predictive model. The basic idea is to consider at any point $\mathbf{x}$ the normal probability density function built with the mean $\mu(\mathbf{x})$ (the prediction) and the standard deviation $\sigma(\mathbf{x})$ (the uncertainty about the prediction) as given by the probabilistic surrogate model. We can virtually build such PDF for any point $\mathbf{x}$: this is illustrated in Figure 8 at $x = 8$, for a one-dimensional problem (notice that, in the case of this figure, instead of the maximum, the minimum of the objective function is pursued). Basically, the area under the tail of the distribution that extends below the line individuated by the f_{min} value in the figure represents the "expected improvement" over the best current known function value. Hence, the improvement at any $\mathbf{x}$ point is given by:

$$I(\mathbf{x}) = max\big(\mathcal{N}\big(\mu(\mathbf{x}), \sigma^2(\mathbf{x})\big) - f(\mathbf{x}^+), 0\big),$$

where $\mathcal{N}\left(\mu(\mathbf{x}), \sigma^2(\mathbf{x})\right)$ is the normal probability distribution given mean $\mu(\mathbf{x})$ and variance $\sigma^2(\mathbf{x})$. It is possible to compute the expected improvement by taking the expected value of $I(\mathbf{x})$:

$$\mathrm{E}[I(\mathbf{x})] \equiv \mathrm{E}\left[max\left(\mathcal{N}\left(\mu(\mathbf{x}), \sigma^2(\mathbf{x})\right) - f(\mathbf{x}^+), 0\right)\right],$$

which can be evaluated by considering the integral:

$$\mathbb{E}(\mathrm{I}) = \int_{\mathrm{I}=0}^{\mathrm{I}=\infty} \mathrm{I} \frac{1}{\sqrt{2\pi}\sigma(\mathbf{x})} \exp\left(-\frac{(\mu(\mathbf{x}) - f(\mathbf{x}^+) - \mathrm{I})^2}{2\sigma^2(\mathbf{x})}\right) d\mathrm{I}$$

This integral can be solved analytically (Jones, Schonlau, & Welch, 1998), yielding the expected improvement acquisition function:

$$\mathrm{EI}\,(\mathbf{x}) = \begin{cases} \sigma(\mathbf{x})\left[\frac{\mu(\mathbf{x}) - f(\mathbf{x}^+)}{\sigma(\mathbf{x})}\Phi\left(\frac{\mu(\mathbf{x}) - f(\mathbf{x}^+)}{\sigma(\mathbf{x})}\right) + \phi\left(\frac{\mu(\mathbf{x}) - f(\mathbf{x}^+)}{\sigma(\mathbf{x})}\right)\right] & \text{if } \sigma(\mathbf{x}) > 0 \\ 0 & \text{if } \sigma(\mathbf{x}) = 0 \end{cases}$$

where $\Phi(\cdot)$ and $\phi(\cdot)$ are the CDF and the PDF of the of the standard normal distribution, respectively. Practically, the expected improvement is higher the lower is the prediction $\mu(\mathbf{x})$ and the higher is the uncertainty $\sigma(\mathbf{x})$, which enables performing the automatic tradeoff between exploitation and exploration.

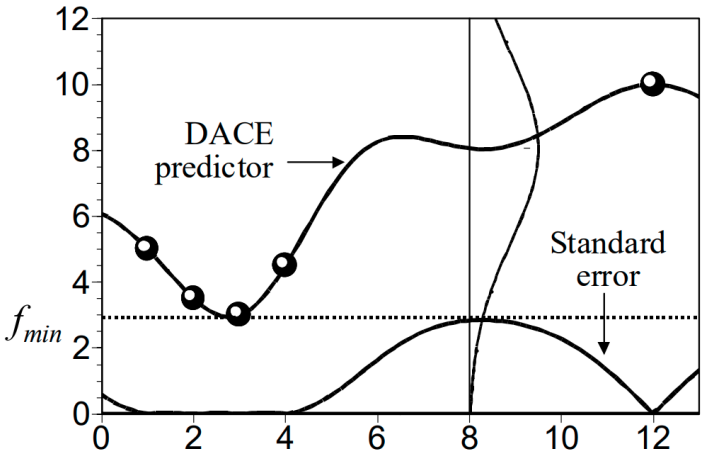


Figure 8. The PDF at $x = 8$ is completely described by the model mean $\mu(\boldsymbol{x})$ and the standard deviation $\sigma(\boldsymbol{x})$. In this figure, taken by *(Jones, Schonlau, & Welch, Efficient Global Optimization of Expensive Black-Box Functions., 1998)*, the stochastic process is called DACE, "Design and Analysis of Computer Experiments".

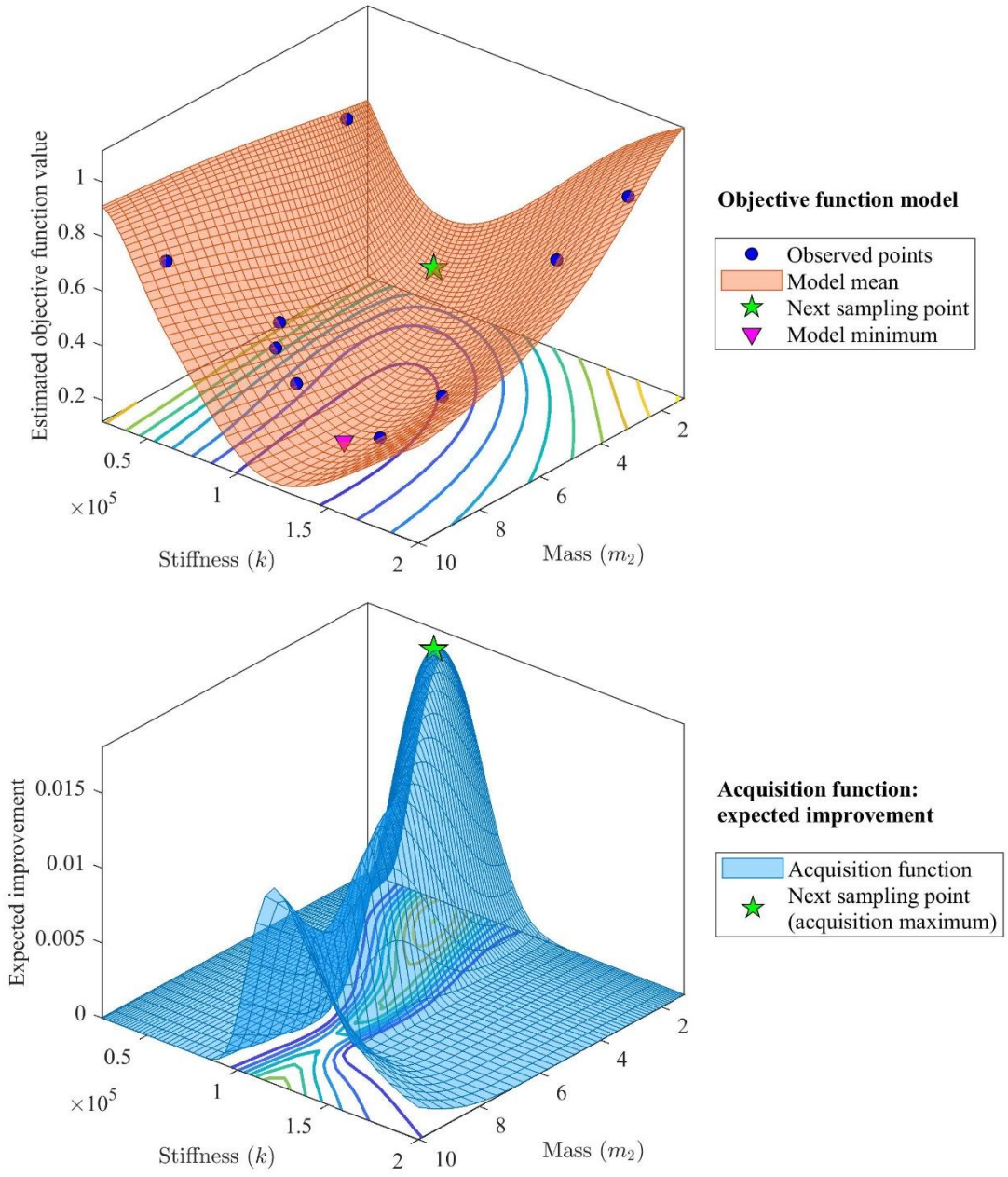


Figure 9. At the top: the plot represents the mean of the GP, $\mu_{1:9}(\boldsymbol{x})$, the nine observations $\boldsymbol{f}_{1:9}$, the model minimum (i.e., the lowest function value as predicted by the GP), and the next sampling point selected by the acquisition function expected improvement (EI). At the bottom: the plot displays the acquisition function $EI(\boldsymbol{x})$ and the next chosen sampling point, which corresponds to the acquisition function maximum. Notice how the maximum of EI is found in a region scarcely populated by observations, which uncertainty about the prediction is high. The selected point it this case is very different from the one chosen by PI: EI shows much better explorative attitudes.

Figure 9 shows the surrogate model (GP) and the EI acquisition function in the framework of a Bayesian optimization approach for the same updating problem of the previous

paragraph. In this case, EI selects a very different point for sampling: the choice is here driven by taking much more into account the uncertainty we have over the surrogate model predictions. In fact, the chosen point is quite far from the actual surrogate model minimum and is found to be in an area far from other observations, which uncertainty is high while the predicted objective is still reasonably low. Hence, EI considers this point to be the one *potentially* leading to the maximum improvement over the incumbent. The use of EI enables a better global search of the optimum: expected improvement is therefore quite suitable when seeking the minimum of highly non-convex functions. Nonetheless, this acquisition function still has a consistent chance to get trapped in a local minimum. A way to reduce this outcome is achievable by increasing the initial seed size to enhance the reliability of the predictive model, at the expense of more function evaluations.

Expected improvement (as in this case) often shows a highly multi modal behavior (Jones, Schonlau, & Welch, 1998). While relatively cheap to compute (since running on top of the surrogate model), multi-peaked acquisition functions can be quite problematic to maximize, increasing the computational burden that comes from the maximization task at this step of the optimization procedure.

2.3.3.3 Expected improvement as suggested by Bull

Typically, for deterministic (noise-free) objective functions, a small quantity of Gaussian noise is added to the observations, so that the total variance σ_T^2 at a given point $\mathbf{x}$ is given by:

$$\sigma_T^2(\mathbf{x}) = \sigma^2(\mathbf{x}) + \sigma_A^2 \,,$$

where σ_A^2 is the additive noise and $\sigma^2(\mathbf{x})$ is, as always, the variance given by the predictive model.

In this variation, EI is used exactly like described in the previous paragraph, but this time an *overexploitation check* is performed at each iteration of the algorithm. The goal is to recognize when the optimization procedure is overexploiting a certain area of the

optimization space and force the algorithm to sample the objective away from this region. A way to achieve this goal is proposed by (Bull, 2011). The following overexploitation check is performed on the selected point $\boldsymbol{x}_{t+1}$ (which corresponds to the acquisition function maximum):

$$\sigma(\boldsymbol{x}_{t+1}) < t_\sigma \sigma_A \,,$$

where t_σ is an arbitrary coefficient, that controls the overexploitation threshold.

If the above condition is true, the kernel is modified by multiplying the hyperparameters $\boldsymbol{\theta}$ by the number of iterations. This leads to a higher variance in the space between observations, leading to a higher acquisition value in these regions. After forcefully raising the variance, a new point $\boldsymbol{x}_{t+1}$ is selected for sampling and the check is performed once more. If the overexploiting condition is met again, the hyperparameters are multiplied by an additional factor of 10. The modified kernel is fitted to the observation, and the checking process continues until the overexploiting condition is no longer met or to a maximum of 5 times.

2.3.3.4 Upper confidence bound (UCB – LCB)

Upper confidence bound (UCB), or, concerning minimization, lower confidence bound (LCB), is based on a very simple yet very effective concept. Presented (Cox & John, 1992) in the "Sequential Design for Optimization" algorithm (SDO), this acquisition function consists in considering a *lower confidence bound* at $\mathbf{x}$, as given by the statistical features of the predictive model. Note that the researchers were concerned with minimization, hence, the LCB function is defined as:

$$\mathrm{LCB}(\mathbf{x}) = \mu(\mathbf{x}) - \kappa\sigma(\mathbf{x}) \,,$$

where κ is typically a positive integer number, which controls the bound width identified by the standard deviation $\sigma(\mathbf{x})$ and therefore the propensity to explore the optimization space. Often, κ is taken equal to 2, so that the confidence bound is about 95% (indeed, this is the value used in the following applications).

When trying to maximize a function, the following definition of upper confidence bound (UCB) follows spontaneously:

$$\mathrm{UCB}(\mathbf{x}) = \mu(\mathbf{x}) + \kappa\sigma(\mathbf{x})\,.$$

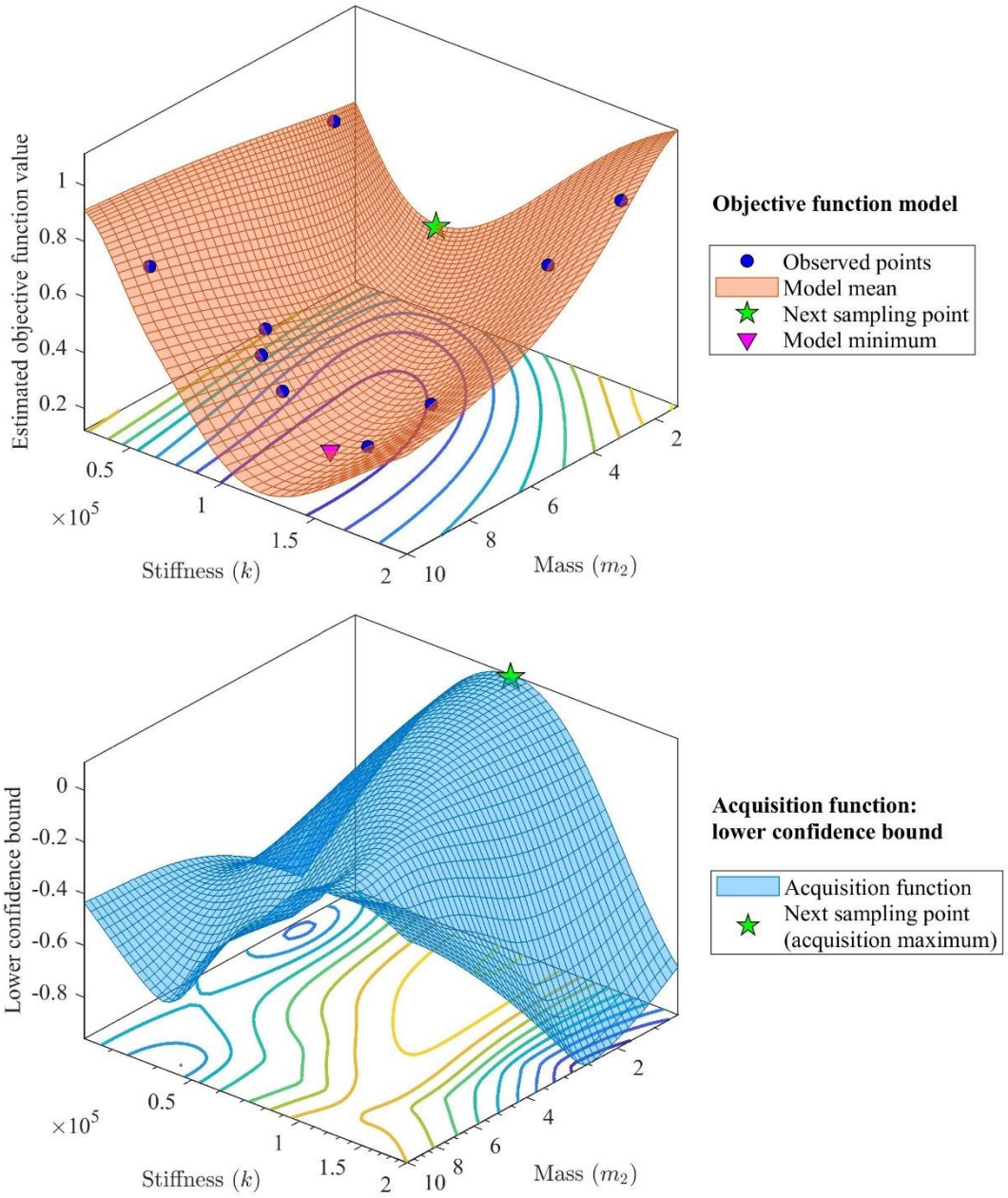


Figure 10. At the top: the plot represents the mean of the GP, $\mu_{1:9}(\boldsymbol{x})$, the nine observations $\boldsymbol{f}_{1:9}$, the model minimum (i.e., the lowest function value as predicted by the GP), and the next sampling point selected by the upper confidence bound (UCB). At the bottom: the plot displays the acquisition function $\mathrm{UCB}(\boldsymbol{x})$ and the next chosen sampling point, which corresponds to the acquisition function maximum. UCB shows an even higher propensity to explore the optimization space, as the next sampling point is chosen farther away from the model minimum.

Figure 10 represents the Gaussian Process and the associated UCB acquisition function, according to the updating problem already considered in Figure 7 and Figure 9. Similar to expected improvement, the next point selected for sampling by UCB is characterized by higher uncertainty. In comparison with EI, upper confidence bound promotes an even more explorative approach for searching the minimum of the objective function: the chosen point is found farther away from the actual predictive model minimum. The impressive aspect of UCB functions is that there is a tendency to explore the optimization space in the first stage of the optimization procedure, when the number of observations is low and the uncertainty high, followed by a more exploitative behavior as the overall uncertainty decreases further on.

3 Details on the algorithm implementation

In this brief section, some technical details about the implementation of the four optimization techniques are given, with an emphasis on Bayesian optimization. As anticipated, the global optimization algorithms that will be employed in various model updating problems are very susceptible to specific implementation choices and to some initial parameters' values (particularly simulated annealing and Genetic Algorithm. For the former, two different strategies will be implemented). As the optimization outcome is affected both in terms of sampling efficiency and accuracy, these aspects must be given appropriate care to enable a meaningful comparison of each algorithm performance.

In Paragraph 3.1, the optimization workflow and the implementation aspects of Bayesian optimization are discussed, while in Paragraph 3.2 details about the other implementation techniques are presented.

3.1 Bayesian optimization implementation details

Technical details about the implementation of the Bayesian optimization procedure, described at the theoretical level in the previous paragraph, are now schematically presented, according to the flowchart represented in Figure 11.

(1) The optimization procedure is initialized by computing the objective function at the seed points, which are randomly chosen within the optimization space, as defined by the optimization bounds. The seed size should be sufficient to avoid overfitting when selecting the optimal set of kernel hyperparameters by maximizing the log likelihood. As a rule of dumb (see Paragraph 2.3.2.2), Jones suggests setting the initial seed size at $10 \cdot d$ at least, where d is the number of dimensions of the optimization problem (i.e., updating parameters). This criterion is followed when applying Bayesian optimization in the next paragraph.

(2) The fitting of the Gaussian Process occurs by maximizing the marginal log-likelihood, which enables to select the optimal set of hyperparameters $\boldsymbol{\theta}^{+}$. Moreover, a small amount of gaussian noise σ^2 is added to the observations (so that the prior distribution has covariance $\boldsymbol{K}(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta}) + \sigma^2 \boldsymbol{I}$).

(3) To maximize the acquisition function, several thousands of predictions $\mu_t(\mathbf{x}_*)$ are computed at randomly chosen points $\mathbf{x}_*$, within the optimization space. Then, some of the best points are further improved with local search (the "fmincon" algorithm is used), among which the best point is finally chosen.

(4) The objective function is computed at the point corresponding to the acquisition function maximum.

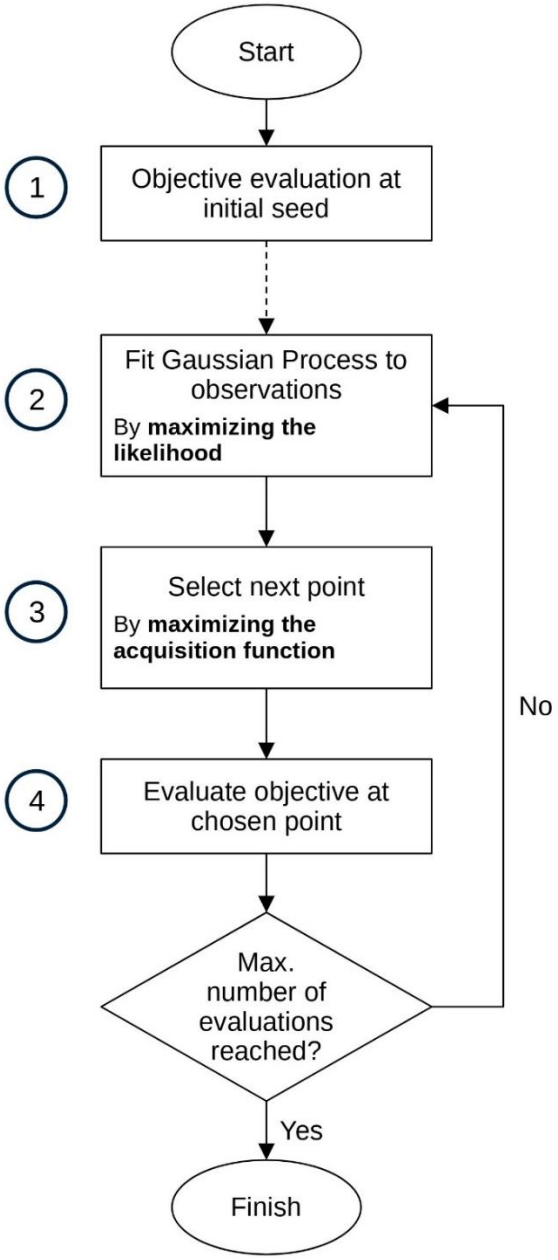


Figure 11. Bayesian optimization procedure workflow.

Before starting the actual procedure (step (2)), some cross-validation tests are performed to determine which configuration of the GP is most suitable for the specific updating problem. First, exhaustive or non-exhaustive (depending on the seed size) cross-validation tests are performed to choose whether to log transform the input variables (i.e., updating parameters), as this is often found to improve the GP regression quality. To this

extent, validation loss is computed for two GPs, one fitted using non-transformed variables, the other fitted using log-transformed variables. Secondly, after choosing weather to transform the input variables or not, the GP is fitted four times using the four kernel functions discussed in Paragraph 2.3.2.1, namely the ARD exponential, the ARD Matérn 3/2, the ARD Matérn 5/2 and the ARD squared exponential kernel functions. Once more, cross-validation loss is used to establish which kernel is the most appropriate at modeling the objective function in analysis.

3.2 GPS, SA, and GA implementation details

GPS algorithm.

- Input parameters are linearly scaled to the interval $[0,100]$, according to the optimization bounds: this is needed since the mesh size is equal in all dimensions.
- The mesh size is multiplied by a factor of 2 at every successful poll, and it is divided by the same factor after any unsuccessful poll.
- The algorithms stops if the maximum allowed number of objective function evaluations is reached, or if the improvement over the last best objective value is lower than 10^{-6}.

Simulated annealing.

Given the susceptibility of the algorithm to different implementation fashions, two different strategies are considered to obtain the results that will be compared to the other optimization techniques in Section 4. The first strategy consists in a very high cooling rate, high initial temperature, and reannealing (the temperature parameter is forcibly increased). The second strategy consists in a lower cooling rate, and slightly lower initial temperature; reannealing is not allowed.

For both strategies, input parameters are linearly scaled to the interval $[0,1]$, according to the optimization bounds. The two different implementations are detailed as follows.

- Strategy 1:
  - The initial temperature T_0 is set at 100.
  - The temperature gradually decreases at each iteration according to the (exponential) cooling schedule $T = T_0 \cdot 0.95^k$, where k is a parameter equal to the iteration number.
  - The reannealing function selects the next point in a random direction, with step-length equal to the current temperature T.

  - Each new sampled point, if its objective is higher than the current one, is accepted according to the acceptance function $\frac{1}{1+\exp\left(\frac{\Delta}{max(T)}\right)}$, where Δ is the difference between the objective values (at the current point and the one just sampled).
  - Reannealing occurs every 100 consecutively accepted sampled points, by artificially lowering the k parameter value.
- Strategy 2:
  - The initial temperature T_0 is set at 50.
  - The temperature gradually decreases at each iteration according to the cooling schedule $T = T_0/k$, where k is a parameter equal to the iteration number.
  - The reannealing function selects the next point in a random direction, with step-length equal to the current temperature T.
  - Each new sampled point, if its objective is higher than the current one, is accepted according to the acceptance function $\frac{1}{1+\exp\left(\frac{\Delta}{max(T)}\right)}$, where Δ is the difference between the objective values (at the current point and the one just sampled).
  - Reannealing never occurs.

Genetic Algorithm.

- Compliance to optimization bounds is enforced by means of two approaches, in the first, constraints are safeguarded through generations, in the second one, optimization bounds are enforced when generating the initial population only. If applying constrains only at the initial population is seen to yield physically meaningful results, the second approach is preferred. Otherwise, the first approach is used.
- The initial population, necessary to initialize the algorithm, consists of points randomly chosen within the space defined by the optimization bounds of each

parameter. The population size should increase accordingly to the numbery of dimensions. The rule of dumb “50 if the number of updating parameters is lower than 5, 200 if higher” is generally applied when using GA in Section 4.

- To choose parents, the following selection criterion is employed: a line divided in sections is considered, each section corresponds to a parent proportionally to its scaled fitness value, according to the scaling function. The line is then divided in segments equal in number to the population size, the amount of nodes laying on one parent section determines its fertility rate.
- The crossover fraction is set at 0.8. Depending on whether the optimization constrains are enforced at the initial population only or throughout the optimization process, two different crossover operators are used. Both use stochastic criteria.
- The elite size is set at 5% of the population size.
- The mutation fraction varies dynamically, according to the genetic diversity at each generation.

4 Assessing the performance of Bayesian optimization in structural dynamics model updating problems

Due to the different nature of the optimization techniques employed, some appropriate practices must be applied to promote a fair comparison of the algorithms, that is necessary to evaluate the performance of the Bayesian optimization approach for model updating applications:

- Since simulated annealing and Genetic algorithm are heuristic optimization techniques, the final optimization results are variable. Hence, several optimization runs are performed with both algorithms: in the following case-studies, the results shown stem from 10 different runs. Either the average or a representative case (e.g., the fourth best-performing run) of the 10 executions are further on considered.
- The GPS algorithm must be initialized from a starting point. Obviously, the distance from the (known, in case of numerical tests) global optimum to the selected starting point greatly impacts the algorithm efficiency and effectiveness. Hence, the algorithm is initialized at an arbitrary chosen point, that is not too close nor too far from the optimum. When appropriate, results of a GPS minimization starting from the extremes of the optimization bounds (upper and/or lower bounds) are displayed to evaluate the eventual detrimental impact on the optimization performance.
- As the performance of simulated annealing is very sensitive to the specific implementation used, two different strategies (according to what detailed in Paragraph 3.2) are employed. The first strategy consists in using a high cooling rate, while reannealing several times during the optimization process. The second strategy makes use of a much lower cooling rate, but reannealing is not permitted.
- To this extent, the following penalty (or cost) function is considered:

Penalty function.

The kind of modal data (i.e., natural frequencies and mode shapes) used for updating varies according to the case-study: for numerical case-studies, *target* natural frequencies and the related mode shapes generated by arbitrarily chosen target parameters will be used; for the experimental case-study (the damage assessment of the Mirandola bell tower), modal data used for updating consists of *identified* natural frequencies and associated mode shapes. For what concerns mode shapes, the MAC value is considered (as already discussed in Paragraph 2.1) to measure the coherence between the computed mode shapes and the target (or identified) mode shapes. The natural frequencies and the MAC values used for updating are arranged together to compute the misfit between the computed response and the target response.

To this extent, the following penalty (or cost) function is considered:

$$P = \sum_{i=1}^{N} \left| \frac{\omega_i^{targ/id} - \omega_i^{calc}}{\omega_i^{targ/id}} \right| + \sum_{i=1}^{N} \left(1 - \text{diag} \left(\text{MAC}\left(\boldsymbol{\phi}_i^{\text{calc}}, \boldsymbol{\phi}_i^{targ/id} \right) \right) \right),$$

where $\omega_i^{targ/id}$ and ω_i^{calc} are respectively the i-th target (or identified) natural angular frequency and the i-th computed natural angular frequency out of the N modes used for updating, and $MAC\left(\boldsymbol{\phi}_i^{\text{calc}}, \boldsymbol{\phi}_i^{targ/id}\right)$ is the MAC (Modal Assurance Criterion) value relative to the i-th computed mode shape $\boldsymbol{\phi}_i^{\text{calc}}$ and the i-th target (or identified) mode shape $\boldsymbol{\phi}_i^{targ/id}$. The first term of the penalty function ensures that the natural frequencies calculated by the model are as close to the target ones as possible, while the second term ensures that the target mode shapes and the computed ones are correlated. MAC values close to 0 suggest little or no correlation between modes, while MAC values close to 1 indicate good correlation. As such, if the computed modal data and target modal data are identical, the (always positive) penalty function P is perfectly minimized at 0, which constitutes the global optimum of the function, provided well-posedness of the updating problem. While this is indeed the case in simulated updating procedures conducted by means of numerical data, when using identified modal properties (i.e., experimental data) the global minimum of the penalty function in the output space may lie quite far from

zero. In fact, as further explained in Paragraph 4.3.2, FE models deficiencies coupled with identified modal data inaccuracies lead to ineluctable misfit between computed and measured modal responses.

Performance metrics.

To assess the performance of Bayesian optimization with respect to GPS, SA and GA as well as to benchmark the four optimization techniques, particular focus will be placed on the accuracy level achieved, in relation to the number of function evaluations performed by the algorithms. As fundamentally different optimization methods are being compared, the allowed number of function evaluation differs from case to case: in all case-studies, GPS, SA and GA are allowed to compute the objective function more times than BO (twice or even ten times more), as these techniques typically require a much greater sampling volume to achieve sufficient levels of accuracy. For what concerns the output space (i.e., modal properties), the relative error between target/identified and computed natural frequencies and the MAC values of the associated mode shapes are reported for each case-study. Similarly, for all optimization techniques and for numerical case-studies only, the relative error between target and estimated parameters will be reported, to assess the accuracy level in the input space (i.e., input parameters).

Moreover, the root mean square relative error (RMSRE) is computed to help measuring the overall response misfit of each algorithm. By taking into account all errors, the RMSRE value (computed for both input space and output space quantities) represents a sort of "final score" about the accuracy reached by each technique. For results in the output space, the RMSRE, the root of the average squared relative errors, is computed as:

$$\mathrm{RMSRE} = \sqrt{\frac{1}{n} \cdot \sum_{i=1}^{n} \Delta X_{\mathrm{rel},i}^2 + \frac{1}{n} \cdot \sum_{i=1}^{n} (1 - MAC_i)^2} \,,$$

where $X_{\mathrm{rel},i}$ is the relative error of the i-th natural frequency, MAC_i is the MAC value of the i-th mode shape and n is the number of modes considered for updating. While, for input space results, the RMSRE is simply given by:

$$\mathrm{RMSRE} = \sqrt{\frac{1}{n} \cdot \sum_{i=1}^{n} \Delta X_{\mathrm{rel},i}^2} \,,$$

where $X_{\mathrm{rel},i}$ is the relative error between the i-th updating parameter and its target value, and n is the number of updating parameters considered.

Finally, in the second and third case-studies, also the total optimization time employed by each algorithm is used as a comparison metric, so to showcase the practical benefits of employing a more efficient optimization procedure, such as Bayesian optimization.

4.1 Numerical case study 1: simple 3-DOF shear-type frame

To assess the functionality of all algorithms, and in particular of Bayesian optimization, a very simple model updating setup is considered in this paragraph.

The underlying dynamic system consists in an undamped three degrees-of-freedom (DOF) shear-type frame: the structure is formed by three levels and two columns per level. The columns at each level are considered identical, and the mass is considered lumped. The shear-type frame is visible in Figure 12, which also shows the associated dynamic system consisting in 3 springs (with stiffness k_1, k_2 and k_3) and 3 masses (m_1, m_2 and m_3). These represent all the system constants, as no damping is here present. The stiffness k_i is equal to the lateral stiffness due to bending of the two columns at the i-th floor (each column is considered fixed at both ends, where only horizontal displacements are allowed), while m_i is the lumped mass of the i-th floor. In the figure the three modes of the frame in consideration are reported as well.

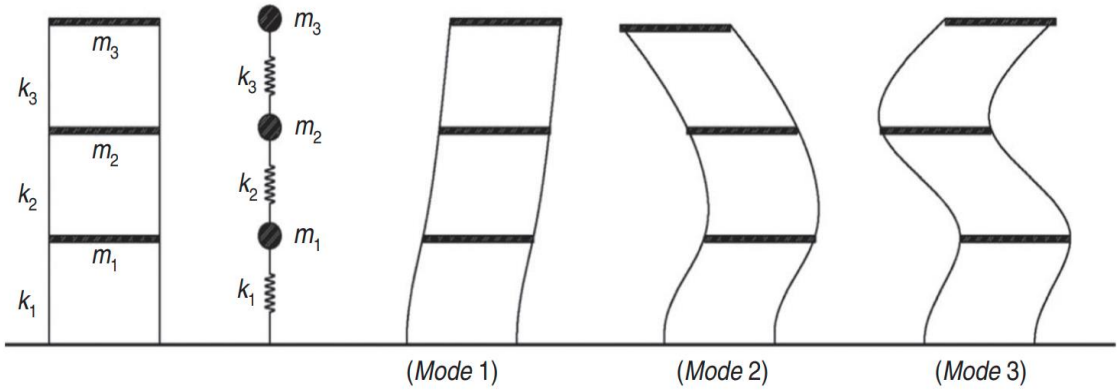


Figure 12. On the left, 3-DOF shear-type frame, along with its associated idealized dynamic system described by the mass constants m_1, m_2 and m_3 and the stiffness constants k_1, k_2 and k_3. On the right, the threes mode shapes of the frame.

The above dynamic system is completely described by the following mass and stiffness matrices:

$$M = \begin{bmatrix} m_1 & 0 & 0 \\ 0 & m_2 & 0 \\ 0 & 0 & m_3 \end{bmatrix} \qquad K = \begin{bmatrix} k_1 + k_2 & -k_2 & 0 \\ -k_2 & k_2 + k_3 & -k_3 \\ 0 & -k_3 & k_3 \end{bmatrix}.$$

As k_1, k_2 and k_3 (which directly constitutes the stiffness matrix elements) will be considered for updating, the geometrical features of the shear-type frame are non-influent, since they never appear in the updating problem.

4.1.1 Model updating setup

The dynamic system described above is used for the model updating procedure set up as follows.

The updating parameters considered are the stiffnesses of the springs at each floor, namely k_1, k_2 and k_3, while the lumped masses m_1, m_2 and m_3 are all considered equal to 6 kg and fixed in value. The modal data (i.e., natural frequencies and mode shapes) used for updating has numerical nature, since it is generated starting from a set of target parameters, taken within the optimization bounds. This *target modal data* consists in the 3 natural frequencies and the associated 3 mode shapes of the dynamic system. For what concerns mode shapes, the MAC value is considered (as already discussed in Paragraph 2.1) to measure the coherence between the computed mode shapes and the target mode shapes. The 3 natural frequencies and the 3 MAC values are arranged together to compute the misfit between the computed response and the target response. To this extent, the penalty function described in Paragraph 4 (Penalty function.) will be considered. If the computed modal data and target modal data are identical, the (always positive) penalty function P is perfectly minimized at 0, which constitutes the global optimum of the function, provided well-posedness of the updating problem.

Updating parameters.

In this model updating setup, the parameters used for updating are summarized in Table 1. The optimization space is delimited by a lower bound and an upper bound, defined for each updating variable, as found in the table.

Updating parameter	Optimization bounds	
	Lower bound	Upper bound
k_1 [N/m]	1,00E+04	1,00E+06
k_2 [N/m]	1,00E+04	1,00E+06
k_3 [N/m]	1,00E+04	1,00E+06

Table 1. Parameters selected for updating and associated optimization bounds.

Table 2 shows instead the chosen target values of the updating parameters (taken within the optimization bounds).

Updating parameter	Target value
k_1 [N/m]	2,00E+04
k_2 [N/m]	7,00E+05
k_3 [N/m]	6,00E+04

Table 2. Target value of each updating parameter.

Modal data for updating.

As mentioned, all three modes of the dynamic system are used in this model updating setup. Table 3 shows the target modal data, in particular the three natural frequencies (mode shapes are not reported here for practical reasons). The target modal data is generated by the target parameters already shown in Table 2.

Mode n.	Freq.^{TARG} [Hz]
1	32,48
2	123,86
3	490,17

Table 3. Target natural frequencies (generated by the set of target parameters).

The updating setup just described does not present any particular ill-posedness concern: only stiffness-related quantities are chosen for updating and the output data used for updating (the three natural frequencies and respective mode shapes) is found to be sufficient to guarantee uniqueness of solution. For instance, note that when adding the mass to the set of updating parameters the problem becomes ill-posed and the solution is not found to be unique anymore.

4.1.2 Characteristics and objectives of the case study

It is clear that this case study considers a model updating problem where a finite element model is not concerned, as in this setup the system matrices, used to solve the Eigen problem, are directly updated. Notice that an iterative model updating method that makes use of a penalty function is surely not the most befitting strategy to deal with such an updating problem. In fact, when directly updating the system matrices (which has actually little engineering feasibility), direct methods should be employed.

Indeed, this numerical case study has the mere purpose of assessing the functionality of the four optimization algorithms and the relative strategies adopted. Furthermore, this very simple case study is used to showcase the effects that some implementation choices have on the Bayesian optimization procedure. In particular, the impact of different choices over the kernel functions used for the Gaussian Process, the effects of transforming the input variables (i.e., updating parameters) and the implications of using

different acquisition functions on the optimization process and its outcome will be investigated.

For what concerns the optimization problem, in this case the minimization of the cost function P is a relatively easy task, this is true due to a number of reasons: (1) the penalty function, which only involves the solution of a rather simple eigenvalue problem, is very cheap to evaluate, (2) only three parameters are being updated, which results in a fairly low-dimensional optimization problem, (3) the optimization bounds are not particularly wide, narrowing the optimization space and (4) the problem presents comparable sensitivity to all the updating parameters. Since all algorithms are expected to accomplish the minimization task with ease, this case study makes a sensible comparison of the algorithms capabilities impossible to accomplish. Nonetheless, being conceptually simple and computationally cheap, it perfectly serves at assessing the functionality of the algorithms and at drawing some preliminary considerations over some aspects of Bayesian optimization.

4.1.3 Model updating results and performance comparison of the algorithms

In the following section, the results of four model updating procedures obtained through the use of each optimization algorithm are presented. This is done for GPS, simulated annealing (SA), Genetic algorithm (GA), and Bayesian optimization (BO). In the last paragraph of this section, the performances of each algorithm are compared: particular focus will be placed on the number of objective function evaluations performed relatively to the achieved accuracy, both regarding the output data (modal properties) and the input data (updating parameters).

In this case study, generalized pattern search, simulated annealing and Genetic Algorithm will be allowed to perform a maximum of 1000 evaluations of the objective function. Bayesian optimization, as it features much better sampling efficiency, is allowed to evaluate the objective function only 100 times, a sensibly lower amount. To deal with

variability of results, simulated annealing and GA are run 10 times; the average results or a representative run out of the 10 executions is further on considered.

4.1.3.1 GPS algorithm results

Unsurprisingly, due to its simplicity and the high convergence rate, GPS is found to be extremely effective at minimizing the cost function of this specific updating problem. Indeed, fast-converging deterministic algorithms like generalized pattern search are particularly suitable for objective functions easy to optimize (being not characterized by many local minima) and cheap to compute. These should therefore be preferred over fancier optimization techniques.

Figure 13 shows the optimization results as well as what is going on during the process. At the top, the best function value is represented at each iteration. As GPS executes a polling procedure at the mesh points for each iteration, the number of total function evaluations is higher than the number of iterations. This is shown in the plot at the middle: as the maximum basis is chosen to determine the mesh of points, the maximum number of function evaluations per iteration is 6. As the algorithm halts the poll as soon as a better objective value is found, the number of function evaluations is typically lower at the beginning of the optimization procedure, where the algorithm quickly moves towards the optimum. Finally, the bottom plot displays the mesh size, as this is continuously expanded and contracted depending on whether the poll is successful or not. Naturally, the mesh size lowers towards the end of the optimization procedure, as the minimum found by the algorithm is being exploited, and the accuracy over the solution increased.

As visible from the results, despite being allowed to compute the objective function up to 1000 times, the algorithm reaches machine precision with only 283 function evaluations, at which the procedure is consequently halted.

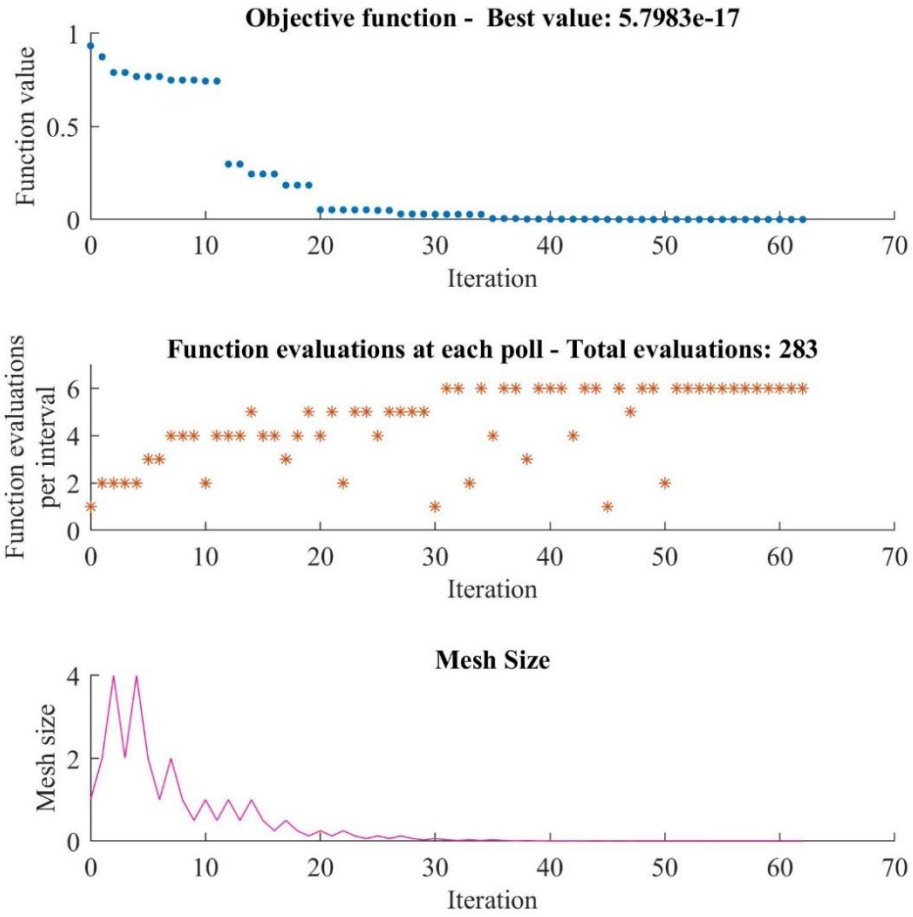


Figure 13. At the top, the objective function value against the number of iterations, and the best objective value achieved. At the middle, the number of function evaluations at each poll against the number of iterations, together with the total number of evaluations computed. At the bottom, the mesh size as it is expanded and contracted throughtout the optimization porcess.

The objective values of all function evaluations are visible in Figure 14. While the algorithm recurrently computes the cost function at less-fit points (which give higher objective values), it quickly and steadily converges towards the global optimum due to the simplicity of the specific optimization problem. While this behavior is welcome in such simple problems, the algorithm can be too “greedy” and get stuck in a local minimum when trying to optimize insidious non-convex functions.

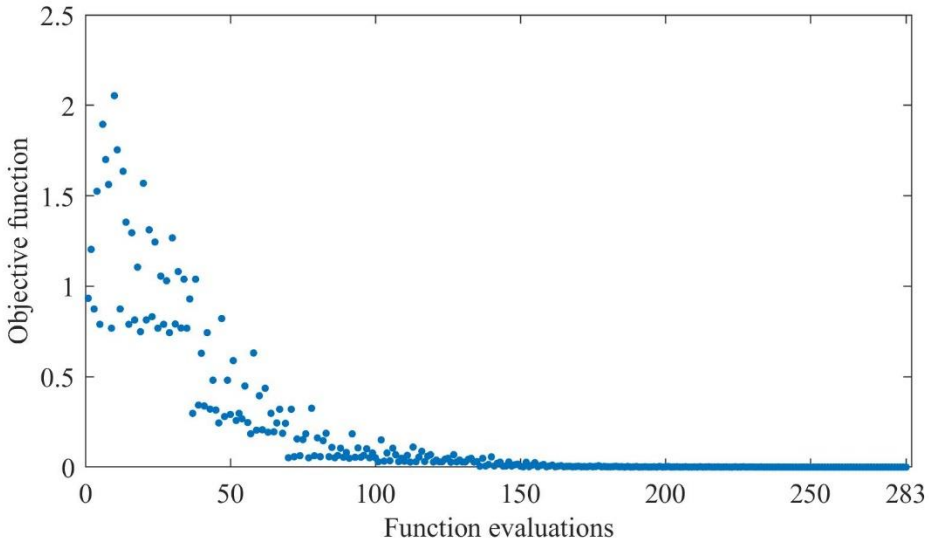


Figure 14. Objective values at each function evaluation executed by the algorithm.

Notice that in this optimization run the algorithm has been initialized at the input point identified by ($k_1 = 1\mathrm{E}+05,\ k_2 = 1\mathrm{E}+05,\ k_3 = 1\mathrm{E}+05$), which is reasonably close to the (known) global optimum. To visualize what happens when initializing GPS from a farther point, the algorithm is run again using ($k_1 = 1\mathrm{E}+06,\ k_2 = 1\mathrm{E}+06,\ k_3 = 1\mathrm{E}+06$) as starting point, which lays at the upper extreme of the optimization space. The results are shown in Figure 15. As GPS is not equipped with multiple length scales, the mesh size is initially kept small to avoid violating the optimization constrains. This, along with the greater initial distance from the optimum, leads to a higher number of functions evaluations before hitting machine precision.

While, as expected, this time more function evaluations are required, the difference with the previous setup is not significant: GPS is again found to be extremely effective, and efficient in terms of sampling.

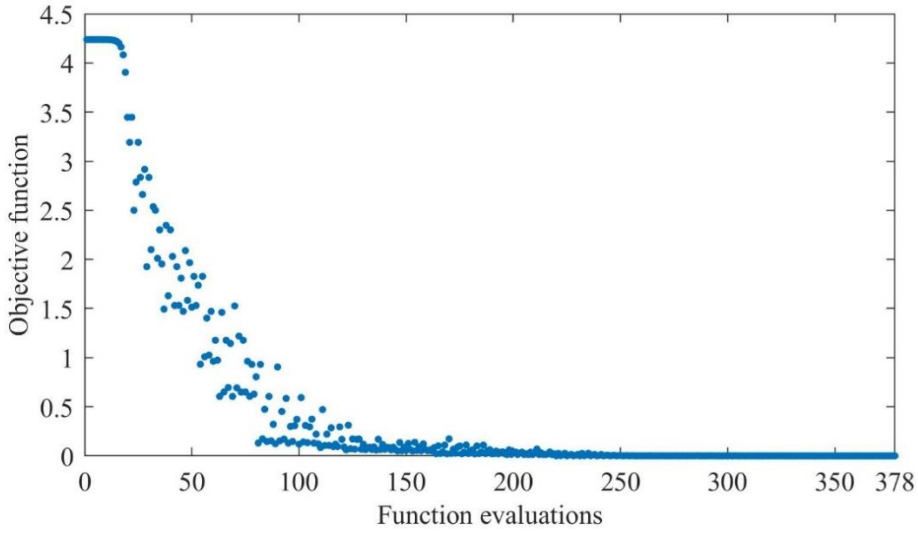


Figure 15. Objective values at each function evaluation executed by the algorithm. GPS is initialized at the upper extreme of the optimization domain.

4.1.3.2 Simulated annealing results

Simulated annealing is a heuristic algorithm. As such, it is executed 10 times to end up with a representative set of results to evaluate its performance in the current optimization problem.

Moreover, since results (in terms of number of evaluations) has been found to be very affected by the specific implementation adopted, two different strategies have been employed. The best performing strategy is used when comparing SA performance to Bayesian optimization and the other techniques.

The first strategy, which for convenience will be referred to as “Strategy 1”, consists in the use of a very fast cooling schedule, recurrent reannealing and a high initial temperature, according to what specified in Paragraph 3.2. In this case study, the initial temperature is set to 100 and the reannealing interval (number of accepted objective function evaluations) is also set to 100. The second strategy, for convenience called “Strategy 2”, entails the use of a slower cooling schedule, no reannealing, and a moderately lower initial temperature. For this strategy, in this case study the initial temperature is set to 50. The input parameters are linearly scaled to the interval [0,1],

according to the optimization bounds (appropriate re-scaling occurs for the optimal parameters at the end of the optimization procedure).

While simulated annealing, similarly to generalized pattern search, must be initialized from a starting point, the particular choices made about the initial temperature and the parameters scaling imply that for the first few iterations (the exact number depends on the initial temperature and the cooling schedule chosen) the algorithm is free to explore any point within the optimization space, with an acceptance probability according to the acceptance function (which is, at the beginning of the procedure, fairly close to 0.5). Hence, the choice of the initialization point is not particularly relevant to the final optimization results in this case.

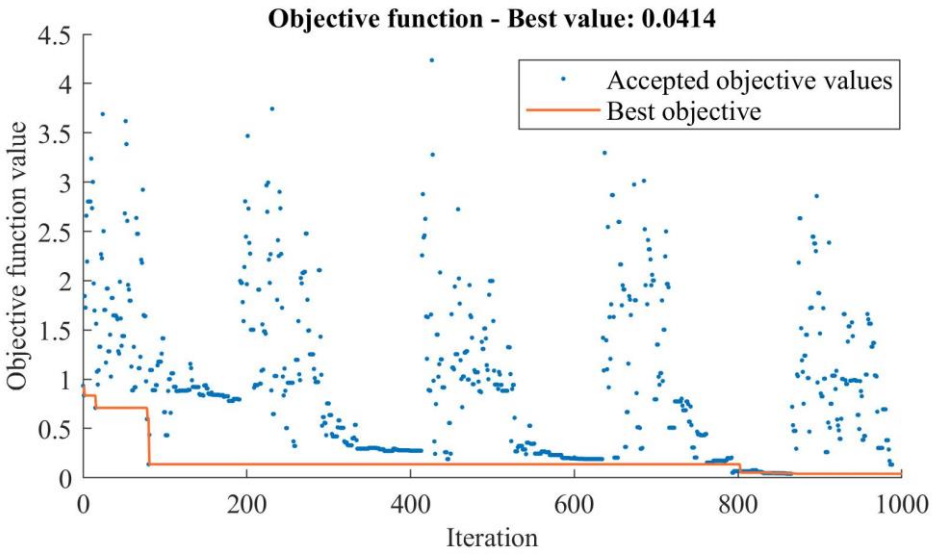


Figure 16. Accepted objective function values and best objective value achieved, plotted against the number of iterations. As visible, using this strategy, reannealing occurs several times during the optimization process.

The results of simulated annealing, using Strategy 1, and its behavior throughout the optimization process are shown in Figure 16. The lowest objective function value, reached after completing 1000 function evaluations, is 0.0414. It is evident how the algorithm reanneals several times, artificially raising the temperature to eventually escape local minima.

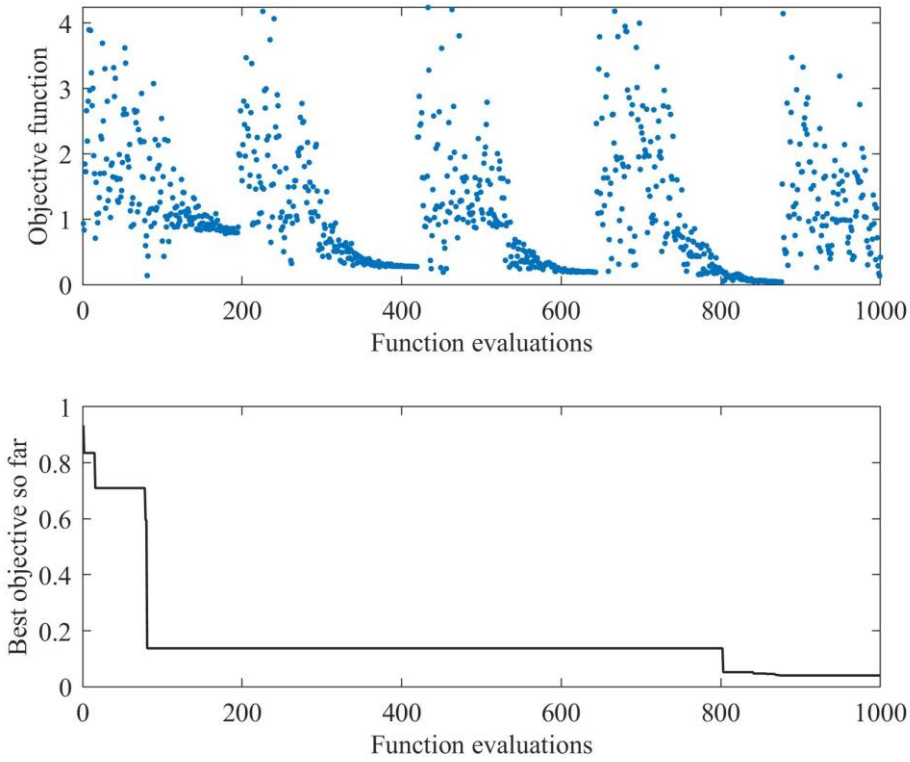


Figure 17. At the top, the objective values of all function evaluations performed by the algorithm during the optimization process. At the bottom, the best achieve value is plotted against the iteration number.

Figure 17 shows the objective values of all function evaluations (also the non-accepted ones) at the top, and the current best objective value as the optimization process continues at the bottom.

Similarly, the results obtained with Strategy 2 are displayed in Figure 18. As the cooling rate is much lower, no reannealing is allowed. Simulated annealing steadily converges towards the objective function minimum. For this strategy, the objective values of all function evaluations and the current best objective value throughout the optimization process are visible in Figure 19.

Considering all 10 optimization runs, Strategy 2 is found to be slightly better performing in terms of accuracy. As such, the optimized modal data and the associated optimized parameters used to compare the other algorithms are obtained using this strategy.

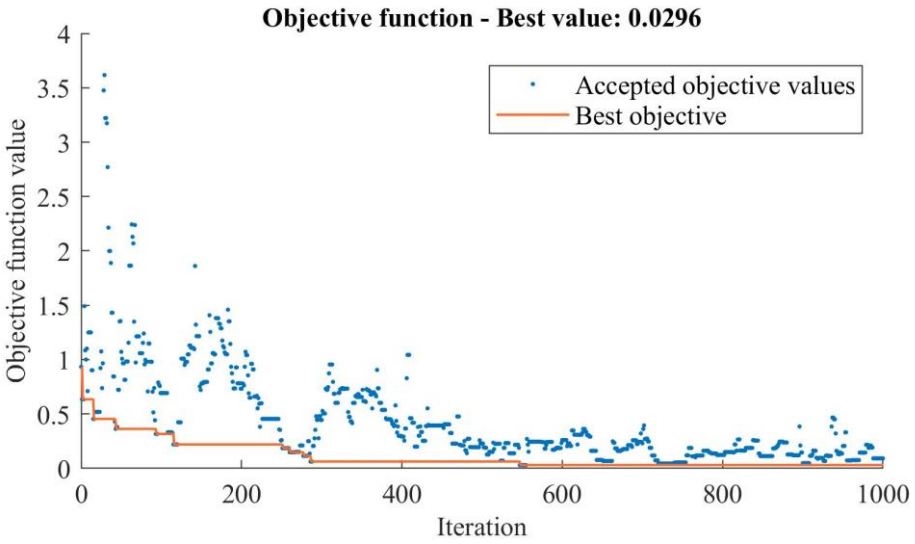


Figure 18. Accepted objective function values and best objective value achieved, plotted against the number of iterations. As visible, using this strategy, no reannealing occurs during the optimization process.

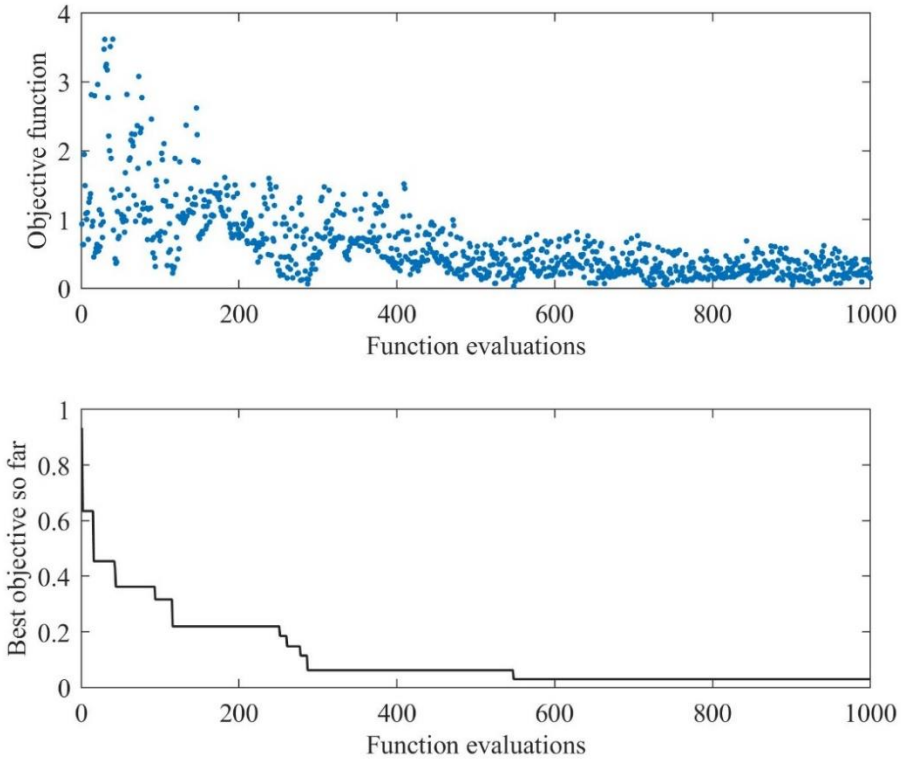


Figure 19. At the top, the objective values of all function evaluations performed by the algorithm during the optimization process. At the bottom, the best achieve value is plotted against the iteration number.

4.1.3.3 Genetic Algorithm results

Like simulated annealing, GA, being heuristic, returns (possibly slightly) different results each time it is run. Hence, the optimization through GA is performed a total of 10 different times for this case study, to end up with a representative set of results that enables to evaluate its performance. Generally, the optimization strategy is set according to what already specified in Paragraph 3.2. In this specific case study, 20 generations starting from an initial population of 50 individuals are considered, resulting in a total of 1000 objective function evaluations.

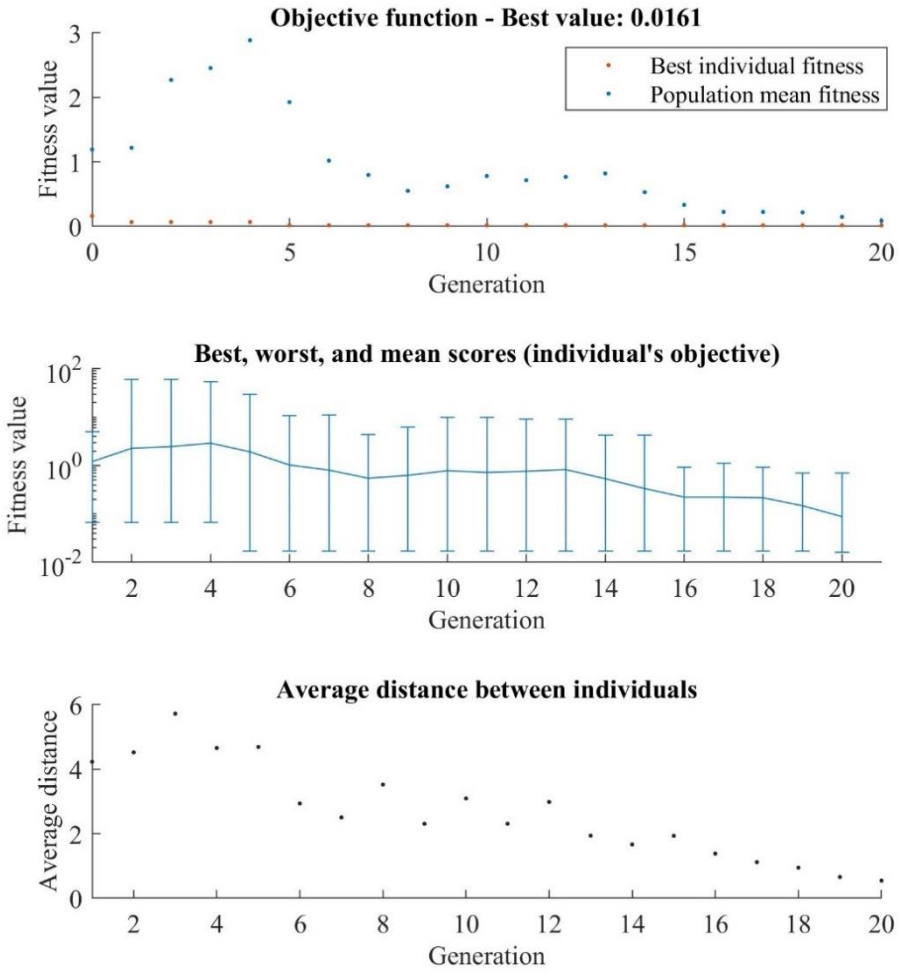


Figure 20. Results of the GA optimization. At the top, the mean fitness value (mean objective of each individual) and the best fitness values (objective of the most-fit individual) are displayed for each generation. At the middle, the best, worst and means objectives of each generation. At the bottom, the average distance between individuals (i.e., the average distance between sampled input points), which represents the genetic diversity of each generation.

The results coming from the third best performing run (in terms of lowest achieved value of the cost function) are visible in Figure 20, which, at the top, shows the best individual fitness (i.e., the best computed objective function value) and the average fitness of the entire population (i.e., the mean of all individual's objective over the entire population, at every generation). The population mean fitness unevenly lowers during the optimization process since due to the stochastic nature of the crossover operator (as well as the mutation operator) employed, children are not guaranteed to be fitter than parent chromosomes. Nonetheless, as fitter individuals have higher chances to breed and as the elite is preserved through generations, the mean fitness gradually improves on the long term. The best objective function value achieved is 0.0161, which denotes a fairly good accuracy given the explorative nature of the algorithm. The plot in the middle further clarifies this, by showing the objective value of the best individual (best score), the objective value of the less-fit individual (worst score) and the mean population fitness (mean score) at each generation. Finally, at the bottom of the figure, the average distance between individuals within each generation is displayed. The average distance represents the genetic diversity of the population, which naturally decreases during the optimization as the algorithm improves accuracy by exploiting the detected (hopefully global) function minimum.

Figure 21, at the top, shows every objective function evaluation during the optimization process. It can be noticed how the algorithm struggles to further improve the best individual beyond the sixth generation. A slight improvement is obtained only in the last generation thanks to one individual (which superior genotype could have potentially led to generalized further improvements in the following generations). Anyhow, the algorithm succeeds at achieving more than satisfactory accuracy. The bottom diagram of the figure in question features the best calculated objective function value throughout the optimization process.

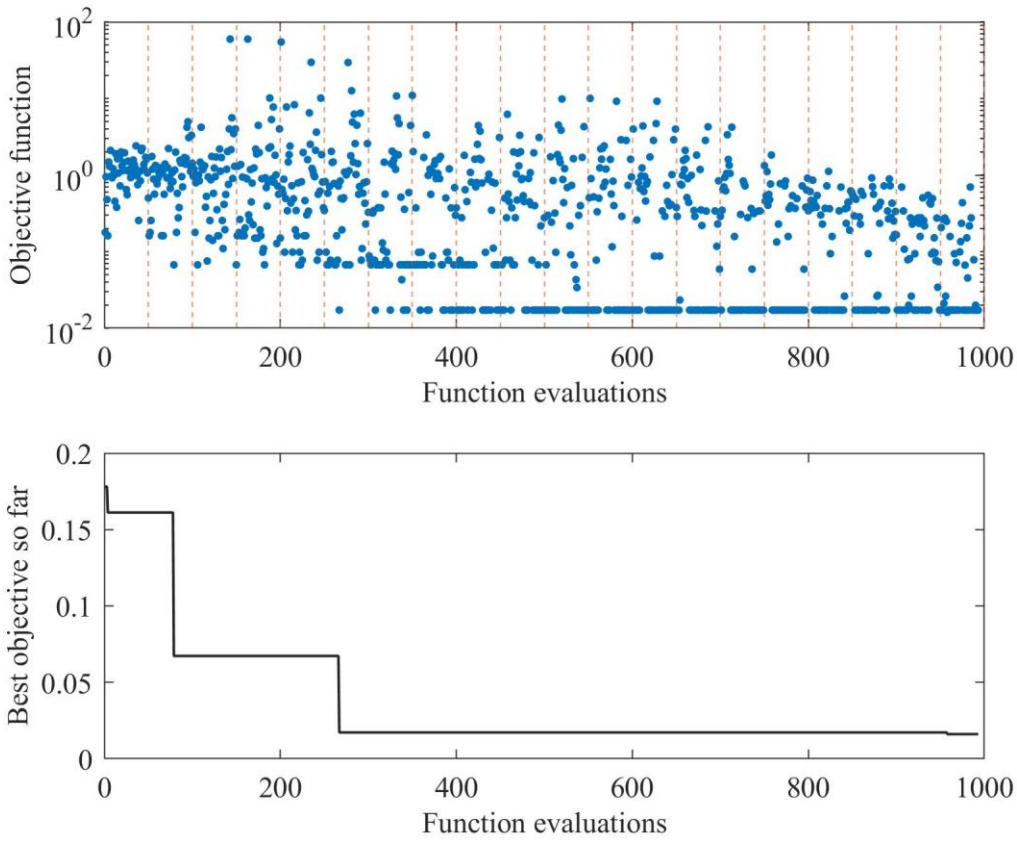


Figure 21. Objective value of all function evaluations performed by GA throughout the optimization process (here, each generation is denoted by the orange dashed line). At the bottom, the best achieved objective value against the number of iterations.

4.1.3.4 Bayesian optimization results

In this paragraph the Bayesian approach is employed to minimize the cost function of the updating problem. After computing the initial seed of points, the effects of transforming the input parameters as well as the impact of using four different kernel functions, namely the exponential kernel, the Matérn 3/2 kernel, the Matérn 5/2 kernel and the squared exponential kernel, will be explored by evaluating the validity of the Gaussian Process through cross-validation techniques. Finally, a Bayesian optimization procedure is run four times to investigate the behavior of the algorithm in terms of convergence speed and sampling efficiency throughout the optimization process when using four different

acquisition functions (i.e., probability of improvement (PI), expected improvement (EI), expected improvement as suggested by Bull (EI+), and lower confidence bound (LCB)).

As mentioned, Bayesian optimization is initialized through an initial seed. The seed is a set of initial observations, obtained by evaluating the objective function at a number of points randomly chosen within the optimization space, as defined by the optimization bounds. The size of the seed is arbitrary, but a commonly accepted rule of dumb is to consider its size equal to at least 10 times the number of dimensions of the problem (see paragraph 2.3.2.2). In this specific case study, where 3 parameters are being updated, this implies that at least 30 points should be considered. Accordingly, the seed size is set to 50 points.

Transformation of optimization variables.

Generally, a logarithmic transformation of the input variables was often found to be beneficial for the surrogate model validation. A better surrogate model validation naturally leads to a smoother optimization process, increased sampling efficiency and a superior final accuracy. Since in this case the initial seed is relatively small, the validity of the surrogate is assessed via an exhaustive cross-validation technique, namely LOOCV, after fitting the Gaussian Process to the initial seed points (at this stage an ARD Matérn 5/2 kernel is used).

Figure 22 shows the results of the leave-one-out cross-validation test. The diagrams on the left are relative to a GP fitted using non-transformed variables, while for the diagrams on the right a logarithmic transformation of the input variables has been employed. The plots on the top show the predicted response against the observed response: the data should lie as close as possible to the 45˚ sloped line for a good model validation. On the bottom, the diagrams display the squared error between the objective value predicted by the GP fitted on the training dataset and the corresponding left-out observation. A slight but non-negligible improvement is observed when log transforming the updating parameters. As such, the following optimizations are performed using log transformed data.

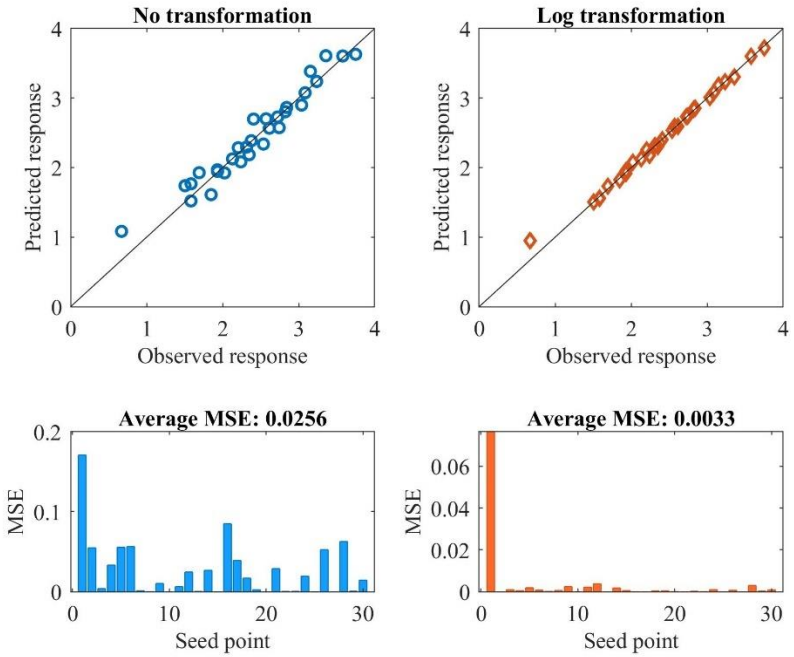


Figure 22. On the left, LOOCV results of a GP fitted using non-transformed variables; on the right, LOOCV results when using log transformed variables. The plots on the top show the predicted response against the observed response: the data should lie as close as possible to the 45˚-sloped line. On the bottom, the diagrams display the squared relative error (SRE) between the objective value predicted by the GP fitted using the training dataset and the corresponding left-out observation.

Kernel functions and parameters' length scales.

Once again, the choice of the kernel function is driven by a cross-validation test, since some kernels may happen to be more suitable at modeling the underlying objective function specific to this updating problem, resulting in surrogate models with enhanced validity. On the contrary, some kernels may lead to a degraded validity of the predictive model, and thus should be discarded.

Table 4 contains the mean squared error given by a LOOCV of four Gaussian Processes, fitted using the ARD exponential, the ARD Matérn 3/2, the ARD Matérn 5/2 and the ARD squared exponential kernels. Generally, all surrogate models are seen to validate quite well. In this case, the ARD Matérn 3/2 kernel is found to be the most suitable at modeling the objective function. However, only a slight difference is appreciable in terms

of cross-validation loss between the kernel functions compared: employing any of the four kernels still most probably leads to a flawless optimization process.

	ARD exponential kernel	ARD Matérn 3/2	ARD Matérn 5/2	ARD squared exponential kernel
MSE [-]	0,0196	0,0007	0,0033	0,0500

Table 4. Cross validation loss when using different kernel function. The mean square error (MSE) is reported for each kernel, averaging the validation losses obtained using LOOCV.

Since automatic relevance determination kernel functions are employed, it is possible to retrieve the length scale of each updating parameter from the hyperparameters of the Gaussian Process (which are determined by maximizing the marginal log-likelihood) fitted to the initial seed of points. The length scales (reported in Table 5) may be seen as some sort of "sensitivity indexes" of the updating variables. As expected, all length scales have a comparable value, since in this case study all the updating variables are of the same nature, meaning the system is about equally sensitive to any chosen input parameter.

Parameter	Length scale [-]
k_1	19,39
k_2	20,37
k_3	22,58

Table 5. Parameters' length scales obtained by maximizing the marginal log-likelihood. All length scales have comparable value. The only slight difference appreciable is that the higher the level, the lower the sensitivity: as expected, stiffnesses corresponding to higher levels have a slightly lower impact on the dynamic response of the system.

Acquisition functions.

After choosing to log transform the input variables and selecting the ARD Matérn 3/2 kernel, the Bayesian optimization process is finally carried out using four acquisition functions. The behavior of the four optimization procedures is visible in Figure 23, where, for each acquisition function used (throughout the optimization process), the objective

value at the randomly sampled seed points is represented in red and the objective value at the points chosen by the algorithm at each iteration is reported in blue.

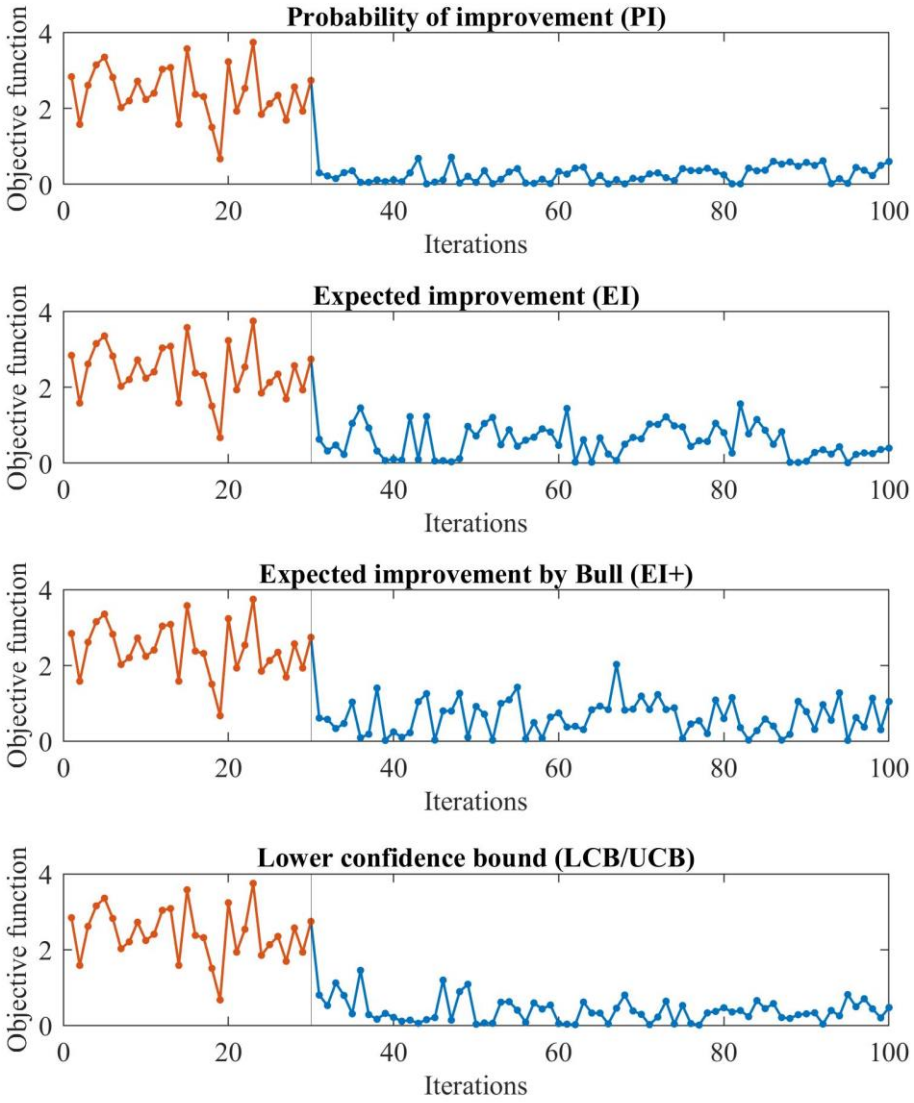


Figure 23. In this figure it is possible to visualize the behavior of the Bayesian optimization throughout the optimization process. The objective value at the randomly sampled seed points is displayed in red, while the objective at points selected by the acquisition function is displayed in blue.

It can be noticed how PI converges much more quickly than the other acquisition functions, that tend instead to further explore the optimization space. This is appreciable also in Figure 24 (that reports the best objective value obtained during the optimization), where it is noticeable the lower value of the first function evaluation when using PI with

respect to the other acquisition functions. While showing a higher propension to explore at early stages, lower confidence bound gradually increases its convergence rate towards the final phases of the optimization process. Indeed, LCB returns the best result in terms of accuracy of the final result (i.e., achieves the lowest cost function value) among the four acquisition functions investigated.

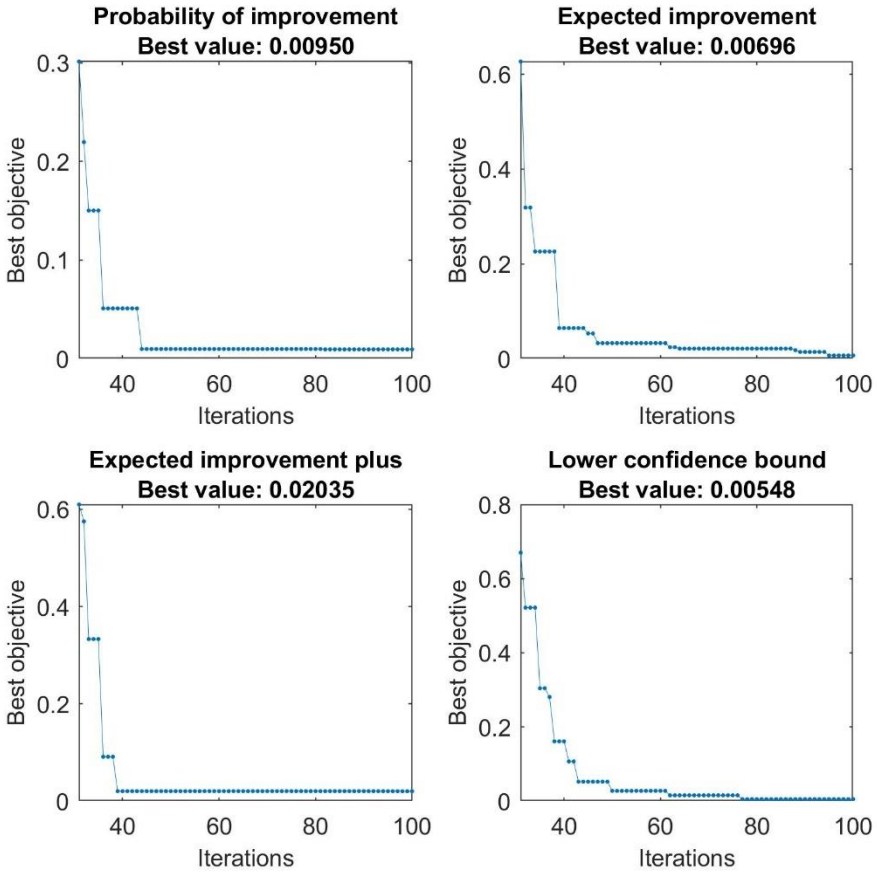


Figure 24. Best objective value achieved during the optimization process when using PI (top left), EI (top right), EI+ (bottom left), and LCB/UCB (bottom right). Notice how the first point selected by PI is lower in value when compared to the other acquisition functions, as this function converges more rapidly. LCB, instead, shows the most gradual minimization, favoring exploration at first and enhancing exploitation at a later stage. As such, LCB is also the one that attains the best objective value.

4.1.3.5 Comparison of optimization performances

Finally, comparing the results obtained with the four optimization techniques enables to evaluate the performance of Bayesian optimization. The results relative to the output, consisting of the raw (best) cost function value and the relative estimated modal data, are shown in Table 6 for each technique. The table reports the relative errors between the updated natural frequencies and the known target values (generated by the set of target parameters), the MAC values (that measure the correlation between estimated mode shapes and target mode shapes), as well as the root mean square relative error (RMSRE), which, by taking into account all errors, represents a sort of "final score" that helps to measure the overall response misfit of each algorithm (refer to Paragraph 4 for more details about its calculation).

As clarified before, this case study, due to the simplicity and cheapness of its cost function, is only suitable to check the functionality of each algorithm. Nonetheless, some comparisons sampling efficiency and some considerations about the implementation aspects of Bayesian optimization (i.e., the effects of transforming the input variables, using different kernel functions and acquisition functions) can already be made. GPS practically shows no errors, as it reaches machine precision accuracy at only 283 function evaluations. Bayesian optimization, when compared to the heuristic algorithms, shows very good performance at minimizing the cost function: with only a tenth of the number of objective evaluations, it reaches a far lower cost function value, which translates in quite a significant improvement over the frequency errors and the MAC values. Simulated annealing and Genetic Algorithm still perform quite well, since both can easily minimize the cost function, even though the accuracy level achieved is less impressive. Again, given the simplicity of the problem, these comes as no surprise.

Mode n.	Freq.^{TARG} (Hz)	Generalized pattern search 1000 fun. eval. Freq.^{UPD} (Hz)	Freq. err. (%)	MAC (-)	Simulated annealing (Strat. 2) 1000 fun. eval. Freq.^{UPD} (Hz)	Freq. err. (%)	MAC (-)
1	32,48	32,48	0,00	1,00	31,13	-4,16	1,00
2	123,86	123,86	0,00	1,00	124,09	0,19	1,00
3	490,17	490,17	0,00	1,00	501,67	2,35	1,00
RMSRE (%)			0,00			1,95	
Best cost f.			0,0000			0,0335	

Mode n.	Freq.^{TARG} (Hz)	Genetic Algorithm 1000 fun. eval. Freq.^{UPD} (Hz)	Freq. err. (%)	MAC (-)	Bayesian optimization (LCB) 100 fun. eval. Freq.^{UPD} (Hz)	Freq. err. (%)	MAC (-)
1	32,48	32,22	-0,82	1,00	32,51	0,09	1,00
2	123,86	107,80	-12,96	1,00	123,06	-0,64	1,00
3	490,17	492,26	0,43	1,00	488,41	-0,36	1,00
RMSRE (%)			5,31			0,30	
Best cost f.			0,0715			0,0055	

Table 6. Optimization results in the output space, obtained with generalized pattern search (GPS), simulated annealing, Genetic Algorithm and Bayesian optimization. For each mode considered, the updated value and the target value are reported, as well as the relative error and the MAC value. Also, the RMSRE and the best achieved objective function value are reported for each algorithm.

The results relative to the input space, and so the updated parameters (that generate the updated modal properties just discussed), are displayed in Table 7. The table reports for each updating parameter, the target value, the updated value, and the relative error obtained using the four optimization techniques.

Updating parameter	Target value	Generalized pattern search 1000 fun. eval.		Simulated annealing (Strat. 2) 1000 fun. eval.	
		Updated value	**Rel. error** (%)	**Updated value**	**Rel. error** (%)
k_1 [N/m]	2,00E+04	2,00E+04	0,05	1,83E+04	-8,65
k_2 [N/m]	7,00E+05	7,00E+05	0,00	7,35E+05	4,93
k_3 [N/m]	6,00E+04	6,00E+04	-0,02	6,04E+04	0,75
RMSRE (%)		0,03		5,77	

Updating parameter	Target value	Genetic Algorithm 1000 fun. eval.		Bayesian optimization (LCB) 100 fun. eval.	
		Updated value	**Rel. error** (%)	**Updated value**	**Rel. error** (%)
k_1 [N/m]	2,00E+04	2,00E+04	-0,25	2,01E+04	0,27
k_2 [N/m]	7,00E+05	7,10E+05	1,49	6,95E+05	-0,71
k_3 [N/m]	6,00E+04	4,45E+04	-25,78	5,92E+04	-1,37
RMSRE (%)		14,91		0,90	

Table 7. Optimization results in the input space, obtained with generalized pattern search (GPS), simulated annealing, Genetic Algorithm and Bayesian optimization. For updating parameter, the updated value and the target value are reported, as well as the relative error between these two. Also, the RMSRE is computed for each algorithm, which represents a "final score" of the optimization results.

Even though all algorithms reach at least decent accuracy, the difference between simulated annealing and Genetic Algorithm, compared to Bayesian optimization, is already quite significative (especially considering that it is only allowed to compute 100 function evaluations, instead of 1000). Notice how a very low value of the cost function leads to an error in the input space that is, though still generally low, already quite significative.

4.2 Numerical case-study 2: expensive FE model of an aluminum structure

The updating problem considered in this numerical case study definitely enables a sensitive comparison of the four optimization techniques. As this case entails a non-trivial and much more expensive cost function in terms of computational complexity, it represents an excellent playground to showcase the capabilities of Bayesian optimization.

The underlying dynamic system is an aluminum frame structure built in the research laboratory of Cranfield University, within the framework of a research focusing on the structural design and performances of a unique optical test system (OTS) used for measuring metre-scale optical surfaces. The structure is well described in the research article "Numerical and Experimental Modal Analysis Applied to an Optical Test System Designed for the Form Measurements of Metre-Scale Optics" (Golano & Zanotti Fragonara, 2018). In this current work, the same (rather complex) finite element model created using the FEA software ANSYS will be used to carry out the numerical model updating procedure. The frame structure is visible in Figure 25, which shows both its CAD model (on the right) and its FE model (on the left). Refer to the original paper for further details about the structure and its FE model, such as the modeling approach, the material properties, the element types, the constrains conditions, etc.

The structure is supported by five leveling feet at the base, that were modeled using numerical artifacts equivalent to 3 orthogonally oriented springs (Figure 26), which stiffness is represented by parameters k_1, k_2 and k_3.

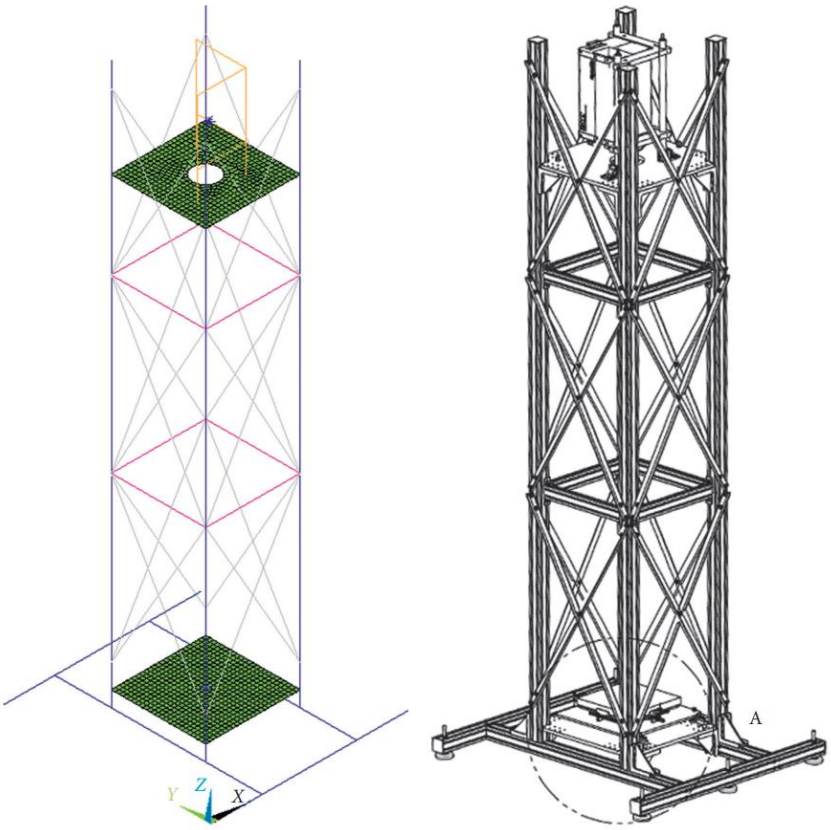


Figure 25. FEA model (left) and CAD model (right) of the aluminum structure used by the researchers. *(Golano & Zanotti Fragonara, 2018)*.

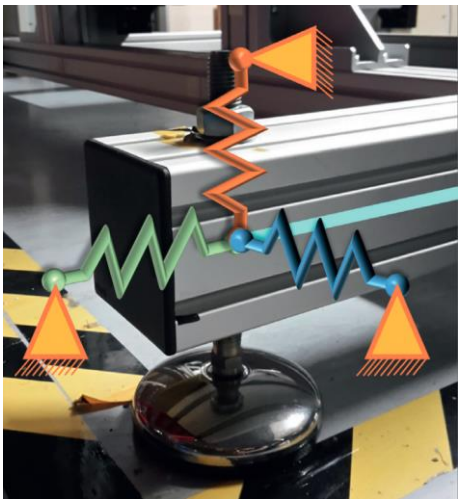


Figure 26. Levelling foot that supported the structure. Superimposed, its modelling approach (three linear springs, k_1, k_2 and k_3, used for the updating procedure). *(Golano & Zanotti Fragonara, 2018)*.

4.2.1 Model Updating setup

The following setup is used for the numerical model updating procedure. The parameters that will be considered are the Young's modulus E_A of the aluminum, which constitutes most of the structure, its Poisson's ratio ν, and the stiffnesses of the three springs that model the five supports, namely k_1, k_2 and k_3, for a total of five updating parameters. The modal data (i.e., natural frequencies and mode shapes) used for updating has numerical nature, since it is generated starting from a set of target parameters, taken within the optimization bounds. This target modal data consists in the first six natural frequencies and the associated six mode shapes of the structure. For what concerns the mode shapes, the MAC value is considered (as already discussed in Paragraph 2.1) to measure the coherence between the computed mode shapes and the target mode shapes. The six natural frequencies and the six MAC values are arranged together to compute the misfit between the computed response and the target response. To this extent, the penalty function described in Paragraph 4 (Penalty function.) will be considered. If the computed modal data and target modal data are identical, the (always positive) penalty function P is perfectly minimized at 0, which constitutes the global optimum of the function, provided well-posedness of the updating problem.

Updating parameters.

In this model updating setup, the parameters used for updating are summarized in Table 8. The optimization space is delimited by a lower bound and an upper bound, defined for each updating variable, as found in the table.

Notice that the optimization bounds are tighter for the aluminum's elasticity modulus E_A, as it is not expected to significantly deviate from the reference value of 70 GPa. Similarly, the optimization range of ν is also limited. The optimization range for the three springs is instead wider (a factor of 5 with respect to the target value, for both the lower and the upper bounds), to represent the higher level of uncertainty that one could have about the rigidity of the supports in a real-case scenario.

Updating parameter	Optimization bounds	
	Lower bound	Upper bound
E_A (GPa)	30,00	150,00
v (-)	0,3	0,4
k_1 (kN/m)	1,20E+03	3,00E+04
k_2 (kN/m)	1,20E+03	3,00E+04
k_3 (kN/m)	1,20E+03	3,00E+04

Table 8. Parameters selected for updating and associated optimization bounds.

Table 9 shows instead the chosen target values for the updating parameters (taken within the optimization bounds).

Updating parameter	Target value
E_A (GPa)	70,00
v (-)	0,32
k_1 (kN/m)	1,00E+04
k_2 (kN/m)	2,00E+03
k_3 (kN/m)	2,00E+04

Table 9. Target value of each updating parameter.

Modal data for updating.

As mentioned, the first six modes of the dynamic system are used in this model updating setup. Table 10 shows the target modal data, in particular the first six natural frequencies (mode shapes are not reported here for practical reasons). The target modal data is generated by the target parameters already shown.

Mode n.	Freq.^{TARG} (Hz)
1	4,39
2	6,50
3	22,70
4	29,15
5	34,57
6	38,75

Table 10. Target natural frequencies (generated by the set of target parameters).

To ensure that the updating problem just described does not suffer ill-posedness, the density of the aluminum was discarded from the updating procedure. The output data used for updating (the first six natural frequencies and respective mode shapes) is found to be sufficient to guarantee uniqueness of solution, as all optimization runs always returned consistent results.

4.2.2 Characteristics and objectives of the case study

The updating problem of this case study is much more complex than the ones seen before, finally enabling to compare the optimization performance of Bayesian optimization with the other techniques. The higher overall complexity of this optimization problem stems from several aspects.

First, the FE model of the frame structure is rather complex: it normally requires about 7 seconds to attain its solution and to compute the first six natural frequencies and mode shapes with an average PC. This may not seem like a lot, but only a few thousands of evaluations translate to several hours of computing time. Therefore, we are dealing with an expensive penalty function to optimize.

Second, the number of updating parameters is not very limited. Five dimensions sensibly increase the complexity of the optimization problem already.

Additionally, as we are optimizing very heterogeneous parameters, namely material properties (like the elasticity modulus and the Poisson's ratio) and springs' stiffnesses, the problem is expected to be sensitive to each updating parameter to a different extent. Hence, the algorithm must be able to deal with such anisotropic optimization problem.

Finally, the cost function of this particular model updating problem was found to be rather insidious, since the global minimum is located in a sharp dip, restrained to a very narrow area of the optimization space. The algorithm has then to be able to spot and exploit this region to achieve a good solution of the optimization problem.

Given the nature of the optimization problem just described, this case study perfectly serves at showcasing the potential of Bayesian optimization, even for what concerns the computational time required to complete the minimization task. In fact, even if fitting the Gaussian Process (by maximizing the likelihood) and selecting the next sampling point (by maximizing the acquisition function) might be computationally intensive tasks (especially as the number of observations increases during the optimization procedure), considered its far superior sampling efficiency, Bayesian optimization potentially enables to save a lot of computational time. Furthermore, just as before, the use of different kernels and acquisition functions will be explored, as well as the benefits of being able to access parameters length scales when using ARD kernel functions.

4.2.3 Model updating results and performance comparison of the algorithms

In the following section, the results of the model updating procedure obtained through the use of each optimization algorithm are presented. This is done for generalized pattern search, simulated annealing (SA), Genetic algorithm (GA), and Bayesian optimization (BO). In the last paragraph of this section, the performances of each algorithm are

compared: particular focus will be placed on the number of objective function evaluations performed relatively to the achieved accuracy, both regarding the output data (modal properties) and the input data (updating parameters), and the total computational time.

In this case study, GPS, simulated annealing and Genetic Algorithm will be allowed to perform a maximum of 200 evaluations of the objective function. Bayesian optimization, as it features much better sampling efficiency, is allowed to evaluate the objective function only 100 times. To deal with variability of results, simulated annealing and GA are now run 5 times; the average results or a representative run out of the 5 executions is further on considered.

4.2.3.1 GPS algorithm results

In this case, generalized pattern search struggles at minimizing the penalty function. The main reasons of sub-optimal results attained by this algorithm are probability to be searched in two problematic aspects: first, GPS does not feature multiple length scales, and second, the optimum is restrained to a very narrow area of the optimization domain. In fact, since the mesh size is the same for all dimensions, GPS behaves well for isotropic problems, but its performance increasingly deteriorates as the difference between the problem sensitivity to each parameter widens. Moreover, given the insidious nature of this particular objective function, if none of the pattern points is ever computed in the neighborhood of the global optimum, the algorithm fails at finding and consequently exploiting the function minimum.

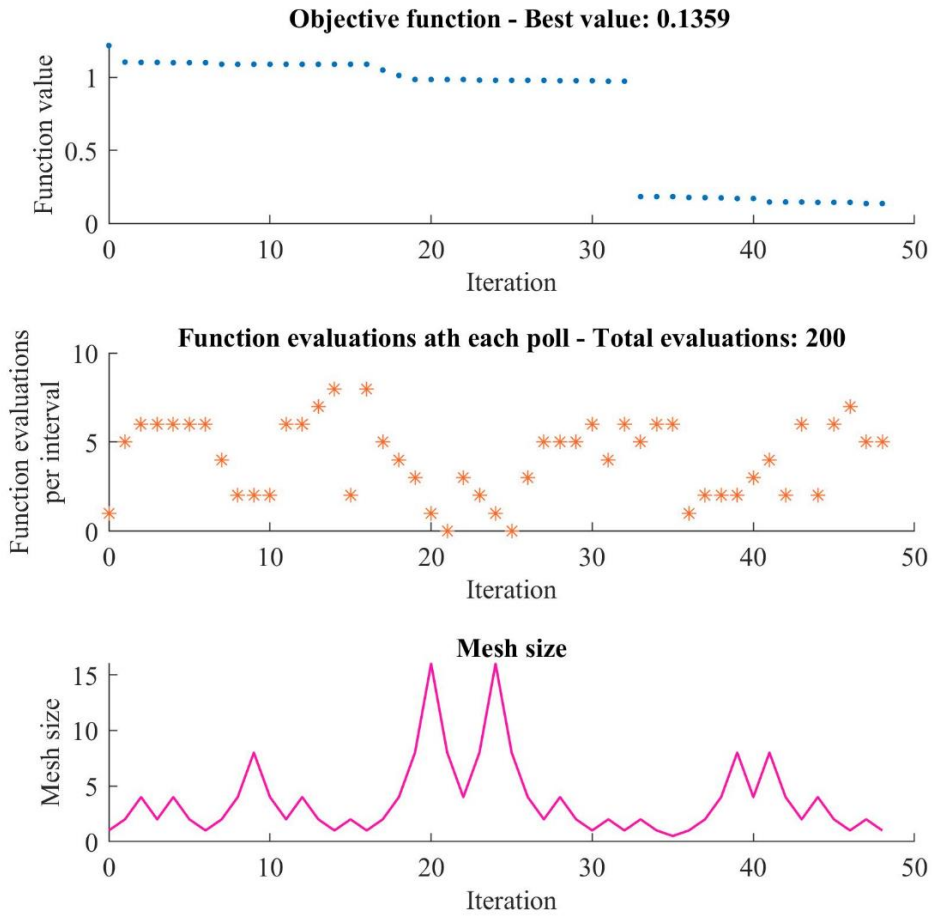


Figure 27. At the top, the objective function value against the number of iterations, and the best objective value achieved. At the middle, the number of function evaluations at each poll against the number of iterations, together with the total number of evaluations computed. At the bottom, the mesh size as it is expanded and contracted throughtout the optimization porcess.

These reasons help to interpret what happens when an optimization procedure through generalized pattern search is initialized at the input point defined by ($E_A = 90,\ \nu = 0{,}36,\ k_1 = 6E+03,\ k_2 = 6E+03,\ k_3 = 6E+03$) – here units of measure are coherent with Table 8 –, which results are shown in Figure 27. At the top of the figure, the best function value is represented at each iteration. As GPS executes a polling procedure at the mesh points at each iteration, the number of total function evaluations is higher than the number of iterations. This is shown in the plot at the middle: as the maximum basis is chosen to determine the mesh of points, the maximum number of function evaluations per iteration is 10. The bottom plot displays instead the mesh size, as this is continuously expanded

and contracted depending on success of the poll. Note that the parameters are linearly scaled with respect to the optimization bound in this optimization run.

As visible from the results, GPS fails at reaching good accuracy (we know that the global optimum of the penalty function is zero), since the best achieved objective value is 0.1359. Such a value is not promising, taking into account that the cost function was already found to be only sensible to consistent changes of input parameters.

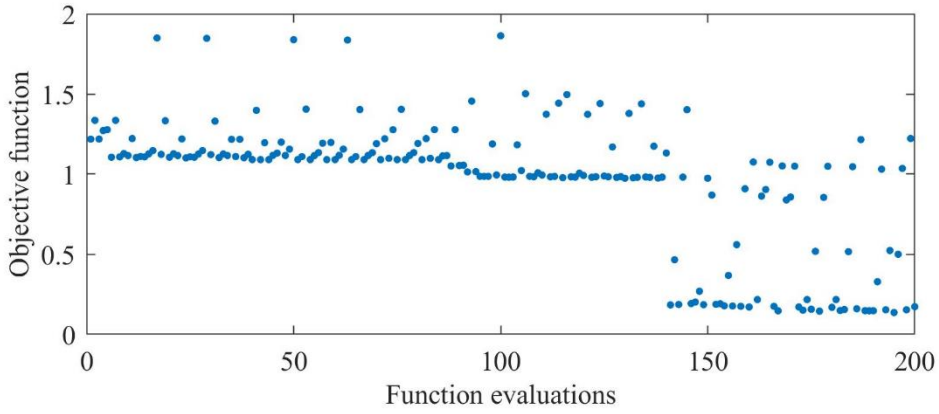


Figure 28. Objective values at each function evaluation executed by the algorithm.

The objective values of all function evaluations are visible in Figure 28. It takes a while for the algorithm (about 140 evaluations) to find objective function values significantly lower than the starting point. Unfortunately, with only 200 evaluations allowed, GPS fails at further minimizing the objective function.

Notice that when starting GPS from a point close to the global optimum (i.e., using a set of updating parameters close to the target one), the results significantly improve. Such initialization points imply a far too optimistic scenario, hence, the results coming from the optimization run displayed above are considered when comparing generalized pattern search to the other techniques.

4.2.3.2 Simulated annealing results

Simulated annealing is a heuristic algorithm. As such, it is executed 5 times (total runs are less than before for convenience, since this case study entails a much more expensive objective function) to end up with a representative set of its optimization performance given the current optimization problem.

For what concerns Strategy 1, in this case study, the initial temperature is set to 100 and the reannealing interval (number of accepted objective function evaluations) is also set to 100. For the second strategy, the initial temperature is set to 50. All differences between the two strategies remain as before. The input parameters are linearly scaled to the interval [0,1], according to the optimization bounds.

As in the previous case, the particular choices made about the initial temperature and the parameters scaling imply that for the first few iterations (the exact number depends on the initial temperature and the cooling schedule) the algorithm is free to explore any point within the optimization space, with an acceptance probability according to the acceptance function (which is, at the beginning of the procedure, fairly close to 0.5 for higher objective values). Hence, the choice of the initialization point is practically non-influent.

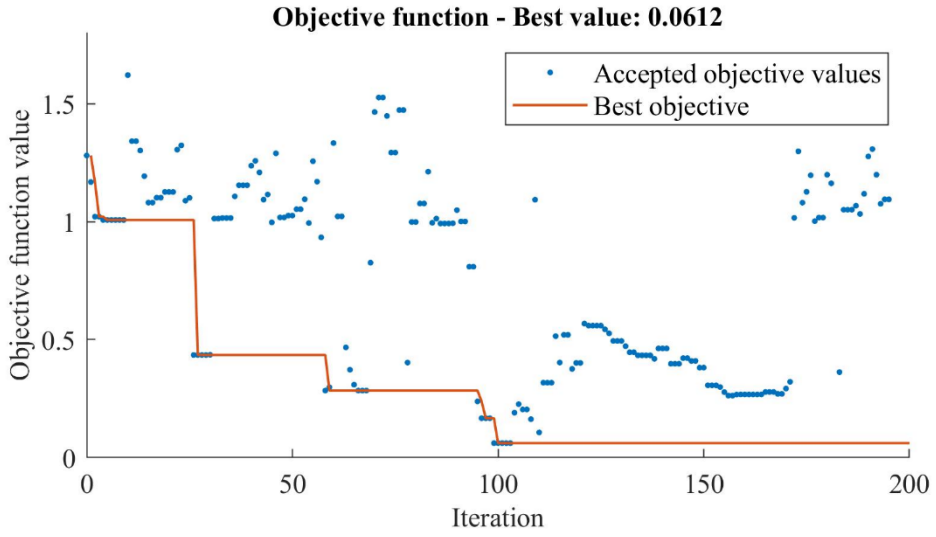


Figure 29. Accepted objective function values and best objective value achieved, plotted against the number of iterations. Using this strategy, reannealing occurs one time during the optimization process.

The results of simulated annealing, using Strategy 1, and the behavior of the algorithm throughout the optimization process are shown in Figure 29. In this specific optimization run, the lowest objective function value, reached after completing 200 function evaluations, is 0.0612.

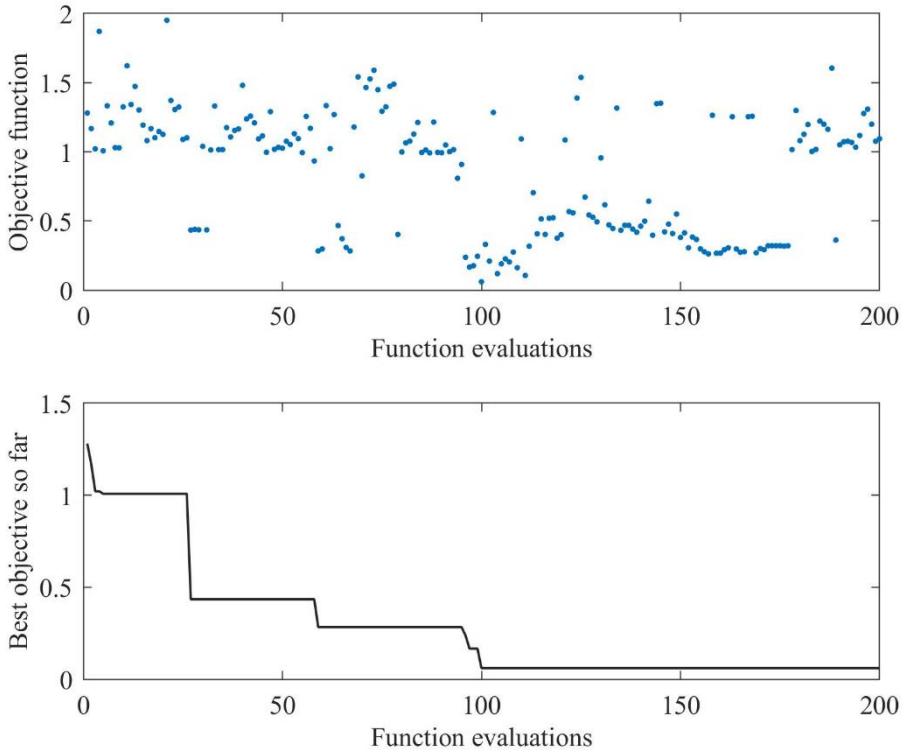


Figure 30. At the top, the objective values of all function evaluations performed by the algorithm during the optimization process. At the bottom, the best achieve value is plotted against the iteration number.

Figure 30 shows the objective values of all function evaluations at the top (also the non-accepted values), and the current best objective value as the optimization process continues at the bottom.

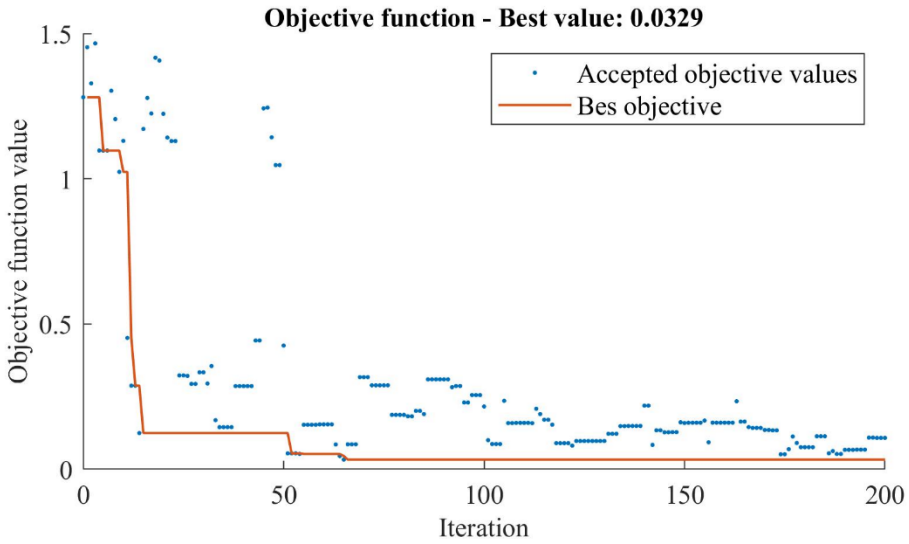


Figure 31. Accepted objective function values and best objective value achieved, plotted against the number of iterations. Using this strategy, no reannealing occurs during the optimization process.

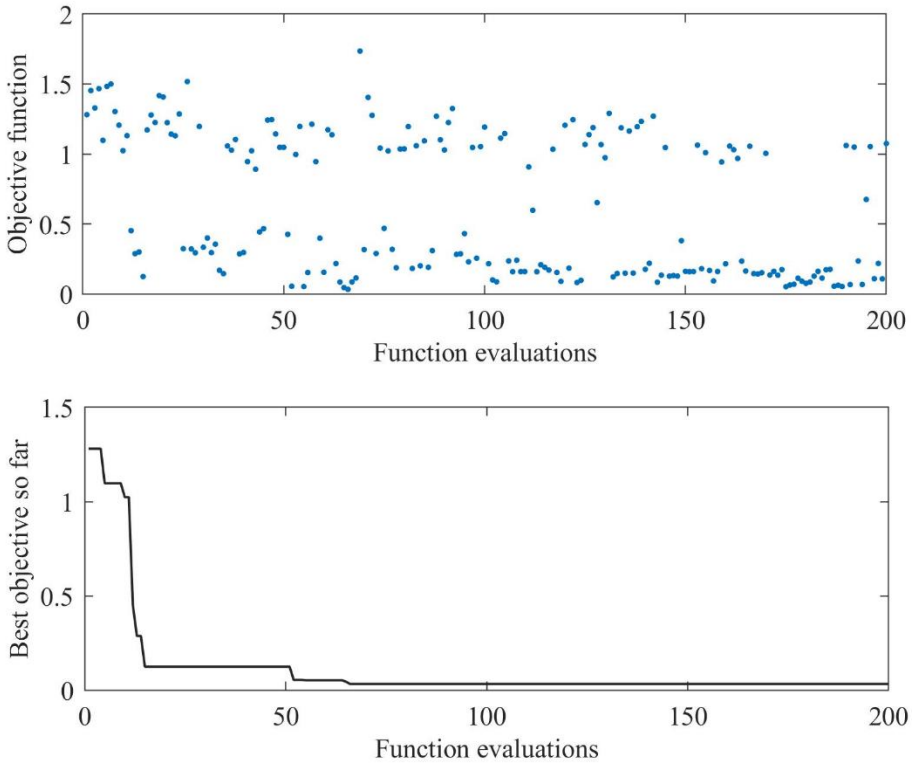


Figure 32. At the top, the objective values of all function evaluations performed by the algorithm during the optimization process. At the bottom, the best achieve value is plotted against the iteration number.

Similarly, the results obtained with Strategy 2 are displayed in Figure 31. As the cooling rate is much lower, no reannealing is allowed. Simulated annealing is seen to steadily converge towards the function minimum.

The objective values of all function evaluations and the current best objective value throughout the optimization process are visible in Figure *32*. During this optimization run, simulated annealing is able to achieve 0.0329 as lowest objective value. Averaging all runs together, Strategy 2 is found to be more capable at minimizing this specific cost function, therefore, the corresponding results will be used when comparing the performance of the four techniques.

4.2.3.3 Genetic Algorithm results

Like simulated annealing, GA, being heuristic, returns different results each time it is run. Hence, the optimization through GA is performed a total of 5 different times for this case study, to end up with a representative set of results that enables to evaluate its performance. In this specific case study, only 4 generations deriving by an initial population of 50 individuals are considered, resulting in a total of 200 objective function evaluations. The results coming from a representative run are visible in Figure 33, which, at the top, shows the best individual fitness (i.e., the best computed objective function value) and the average fitness of the entire population (i.e., the mean of all individual's objective, over the entire population at every generation). The population mean fitness gradually improves from one generation to another. The best objective function value achieved is 0.0517. The plot in the middle shows the objective value of the best individual (best score), the objective value of the less-fit individual (worst score) and the mean population fitness (mean score) at each generation. Finally, at the bottom of the figure, the average distance between individuals within each generation is displayed. The average distance represents the genetic diversity of the population, which naturally decreases during the optimization as the algorithm improves accuracy by exploiting the detected function minimum.

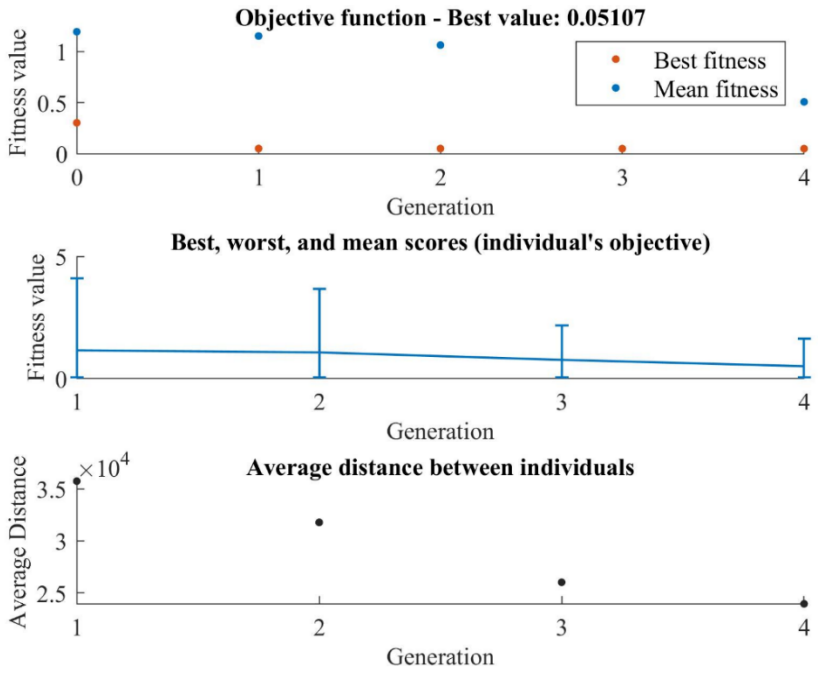


Figure 33. Results of the GA optimization. At the top, the mean fitness value (mean objective of each individual) and the best fitness values (objective of the most-fit individual) are displayed for each generation. At the middle, the best, worst and means objectives of each generation. At the bottom, the average distance between individuals (i.e., the average distance between sampled input points), which represents the genetic diversity of each generation.

Figure 34, at the top, shows every objective function evaluation during the optimization process. The bottom diagram of the figure in question features the best calculated objective function value throughout the optimization process. The algorithm fails at further improving the accuracy after the second generation, despite the population mean fitness is gradually improving throughout the optimization process.

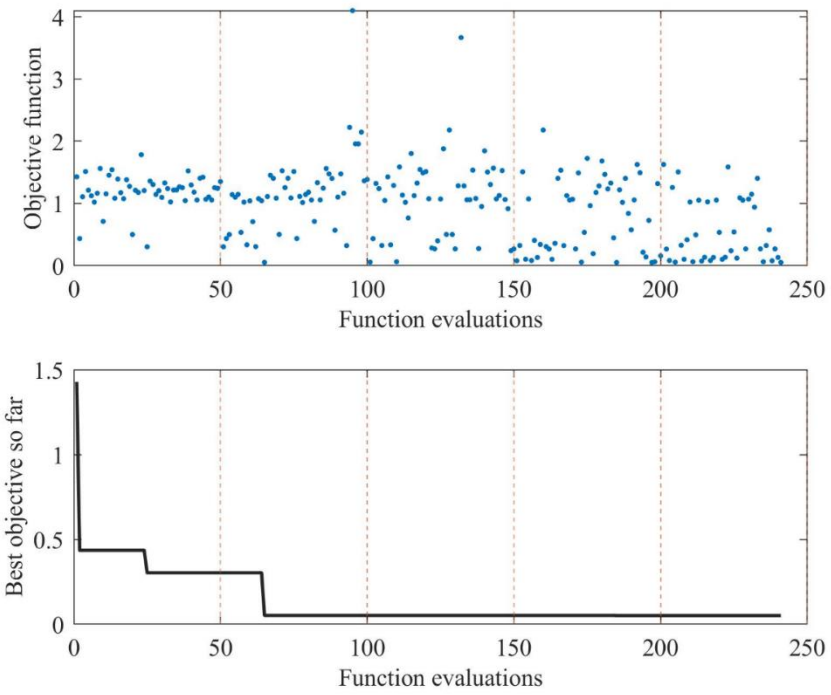


Figure 34. Objective value of all function evaluations performed by GA throughout the optimization process (here, each generation is denoted by the orange dashed line). At the bottom, the best achieved objective value against the number of iterations.

4.2.3.4 Bayesian optimization results

In this paragraph the Bayesian approach is employed to minimize the cost function of the current updating problem. After computing the initial seed of points, the effects of transforming the input parameters as well as the impact of using four different kernel functions, namely the exponential, the Matérn 3/2, the Matérn 5/2 and the squared exponential kernels, will be explored by evaluating the validity of the Gaussian Process through cross-validation tests. Finally, the Bayesian optimization procedure is run four times to investigate, throughout the optimization process, the behavior of the algorithm in terms of convergence speed and sampling efficiency when using four different acquisition functions (i.e., probability of improvement (PI), expected improvement (EI), expected improvement as suggested by Bull (EI+), and lower confidence bound (LCB)).

The seed is obtained by evaluating the objective function at points randomly chosen in the optimization space, as defined by the optimization bounds. Following the usual rule of dumb, in this specific case study, where 5 parameters are being updated, the initial seed size is set to 50 points.

Transformation of optimization variables.

Also in this case, a logarithmic transformation of the input variables is investigated. Since now the cost function is quite expensive, the validity of the surrogate is assessed via a non-exhaustive cross-validation technique, namely a k −fold loss techinique, after fitting the Gaussian Process to the initial seed points (at this stage an ARD Matérn 5/2 kernel is used).

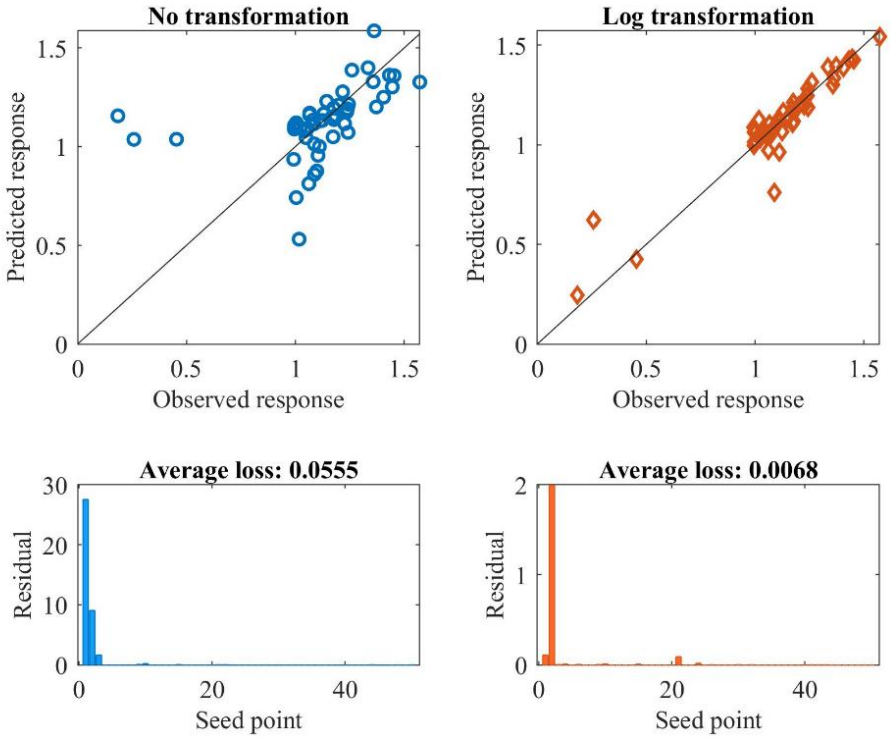


Figure 35. On the left, 10-folds CV results of a GP fitted using non-transformed variables; on the right, 10-folds CV results when using log transformed variables. The plots on the top show the predicted response against the observed response: the data should lay as close as possible to the 45˚-sloped line. On the bottom, the diagrams display the mean squared relative error (MSRE) between the objective values predicted by the GP fitted using the training dataset and the corresponding left-out observations.

Figure 35 shows the results of a 10-folds cross-validation test. The diagrams on the left are relative to a GP fitted using non-transformed variables, while for diagrams on the right a logarithmic transformation of the input variables has been employed. The plots on the top show the predicted response against the observed response: the data should lie as close as possible to the 45˚-sloped line to ensure good model validation. On the bottom, the diagrams display the mean squared error between the objective values predicted by the GP fitted on the training dataset and the corresponding left-out observations. A significant improvement is appreciable when log-transforming the updating parameters. As such, log-transformed variables will be employed for the updating procedure.

Kernel functions and parameters' length scales.

Once again, the choice of the kernel function to be used is driven by a cross-validation test, since some kernels may happen to be more suitable at modeling the underlying objective function specific to this updating problem, resulting in surrogate models with enhanced validity.

	ARD exponential kernel	ARD Matérn 3/2	ARD Matérn 5/2	ARD squared exponential kernel
MSE [-]	0,0091	0,0079	0,0041	0,0072

Table 11. Cross validation loss when using different kernel function. The mean square error (MSE) is reported for each kernel, averaging the validation losses obtained using LOOCV.

Table 11 contains the mean squared error given by a 10-fold cross-validation test of four Gaussian Processes, fitted using the ARD exponential, the ARD Matérn 3/2, the ARD Matérn 5/2 and the ARD squared exponential kernels. Generally, all surrogate models are seen to validate quite well. In this case, the ARD Matérn 5/2 kernel is found to be the most suitable at modeling the objective function. Anyhow, it remains clear that employing any of the four kernels would most probably lead to smooth optimization processes.

Since automatic relevance determination kernel functions are employed, it is possible to retrieve the length scale of each updating parameter from the hyperparameters of the

Gaussian Process fitted to the initial seed of points. The length scales (reported in Table 12) provide information about the sensitivity of the problem to the updating variables; note that a high length scale suggests low sensitivity, while a low length scale indicates high sensitivity. In this case the five length scales don't have the same order of magnitude. In fact, the parameter which majorly affects the problem (i.e., the system modal response) is the aluminum elasticity modulus: such behavior is reasonable, as material rigidity is usually blamed to significantly impact modal properties. On the contrary, and quite reasonably, the problem is not very sensitive to changes of the Poisson's ratio ν. Again, this makes perfect sense as damping affects modal properties only at a marginal level. For what concerns the three springs, k_2 is found to have the greater impact on the system modal response, while k_1 and k_3 feature lower sensitivity. Notice that the length scales, computed by maximizing the likelihood (see Paragraph 2.3.2), are relative to the 50 randomly sampled points: since the problem sensitivity to the updating parameters varies across the input space, slightly different length scales may be obtained when changing the location of the initial seed points.

Parameter	Length scale [-]
E_A	0.59
ν	6239.04
k_1	23.28
k_2	0.98
k_3	8.29

Table 12. Parameters' length scales obtained by maximizing the marginal log-likelihood. As expected, length scales have different value. The parameter that shows the highest sensitivity is the aluminum elasticity modulus E_A.

Acquisition functions.

After choosing to log transform the input variables and selecting the ARD Matérn 5/2 kernel, the Bayesian optimization process is finally carried out using the four acquisition functions. Each optimization run is visible in Figure 36, where, for each acquisition

function used and throughout the optimization process, the objective value at the randomly sampled seed points is represented in red and the objective value at the sampling point chosen by the algorithm at each iteration is reported in blue.

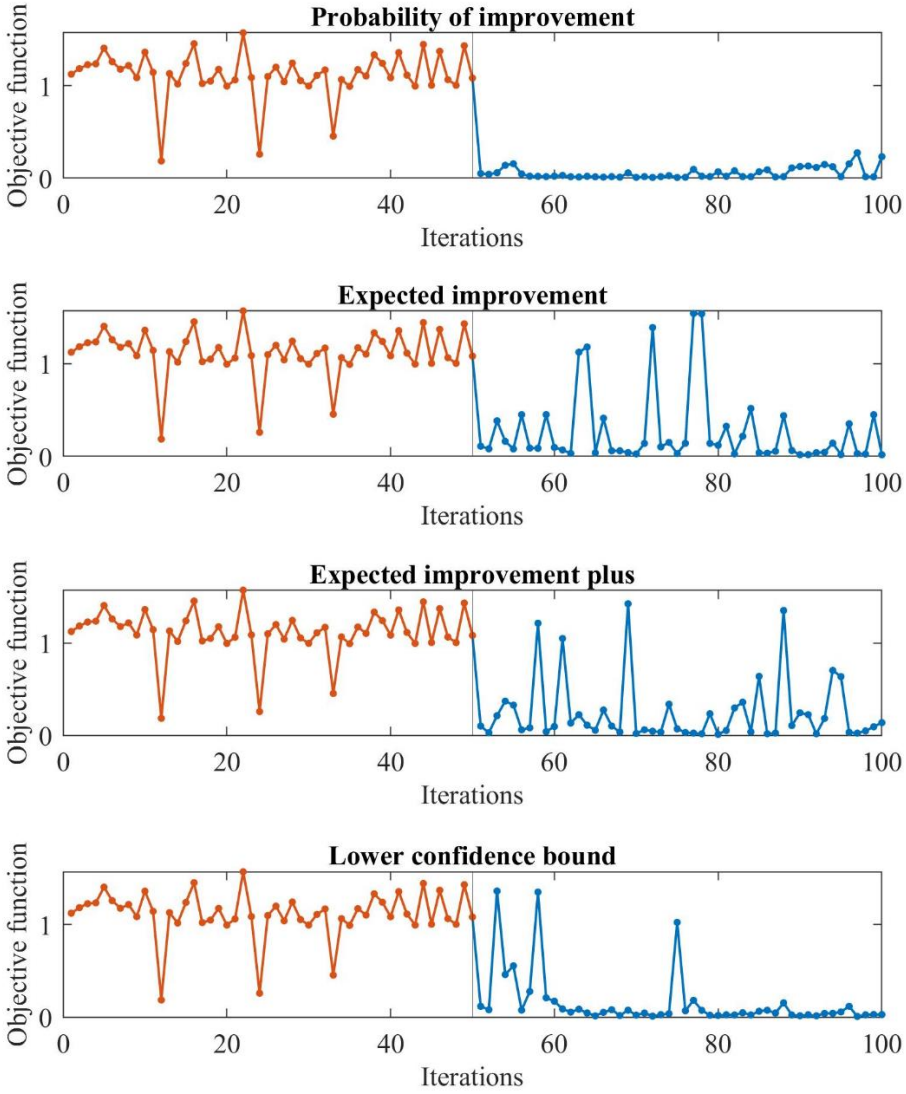


Figure 36. In this figure it is possible to visualize the behavior of the Bayesian optimization throughout the optimization process. The objective value at the randomly sampled seed points is displayed in red, while the objective at points selected by the acquisition function is displayed in blue.

It can be noticed how also in this case PI converges much more quickly than the other acquisition functions, which instead tend to further explore the optimization space. This is appreciable also in Figure 37 (that reports the best objective value obtained during the optimization), where it is noticeable the lower value of the first function evaluation

performed by PI with respect to the other kernels. Given the excellent model validation assessed at the previous step, even though behaving very greedily, PI returns very high accuracy with an extremely limited amount of function evaluations. On the other hand, while showing a higher propension to explore at early stages, lower confidence bound gradually increases its convergence rate towards the final phases of the optimization process. Once more, LCB returns the best result in terms of accuracy of the final solution (i.e., achieves the lowest cost function value) among the four acquisition functions investigated.

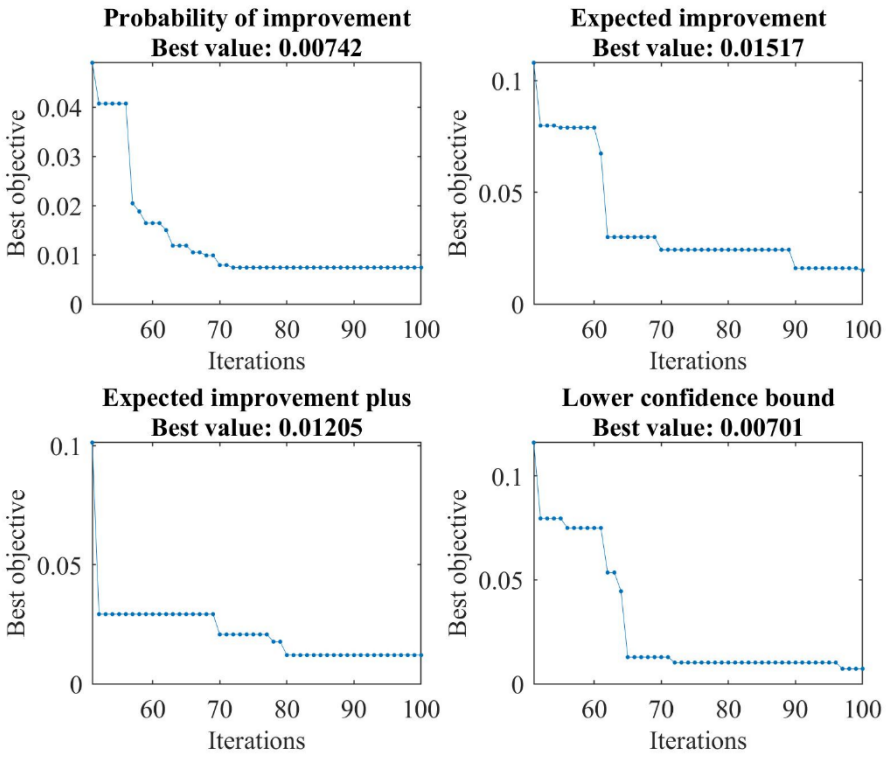


Figure 37. Best objective value achieved during the optimization process when using PI (top left), EI (top right), EI+ (bottom left), and LCB/UCB (bottom right). Notice how the first point selected by PI is lower in value when compared to the other acquisition functions, as this function converges more rapidly. Also in this case, LCB is the acquisition function that attains the best objective value.

Optimization time.

For what concerns the optimization time, since the number of total observations is quite limited, the time required to run the algorithm (modeling and point selection time) marginally impacts the total optimization time, which mostly depends on the time required to evaluate the objective function. This is visible in Figure 38, which shows the total optimization time against the number of function evaluations, as the sum of the objective evaluation time and the modeling and point selection time (which is required to maximize the log likelihood and to maximize the acquisition function).

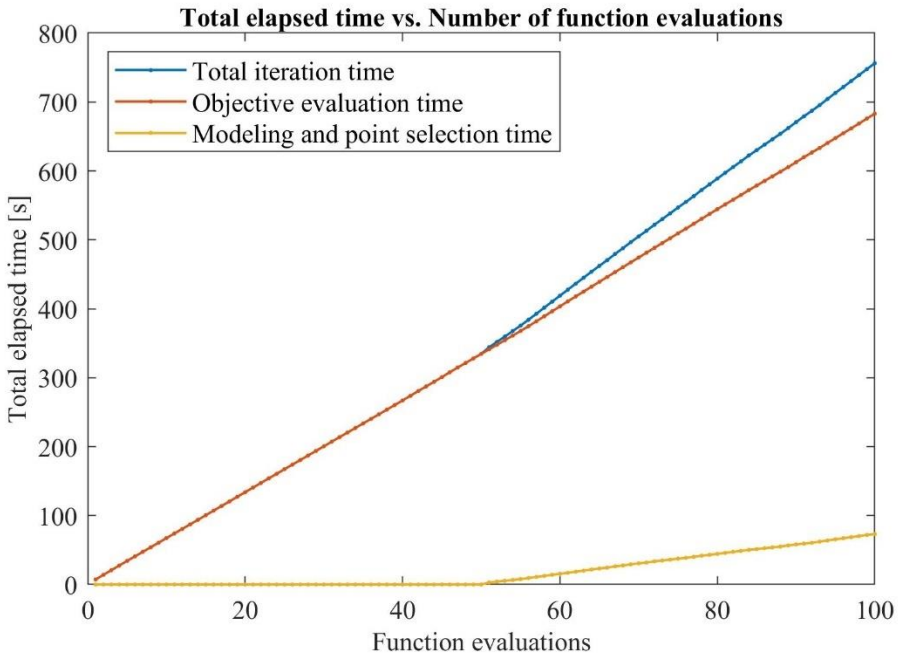


Figure 38. The figure displays the total optimization time as the sum of the objective evaluation time (the cumulative time employed for evaluating the objective function) and the modeling and point selection time (the cumulative time employed for solving the secondary optimization problems, i.e., the maximization of the marginal log-likelihood and the maximization of the acquisition function). As computing a prediction has a computational complexity of $\mathcal{O}(N^3)$, the modeling and point selection time gradually increases as the number of observations N grows.

4.2.3.5 Comparison of optimization performances

Finally, the comparison of the results obtained from the four optimization techniques enables to evaluate the performance of Bayesian optimization. The results relative to the output, that are the raw (best) cost function value and the relative estimated modal data,

are shown in Table 13, for each technique. The table reports the relative errors between the updated natural frequencies and the known target values (generated by the set of target parameters), the MAC values (that measure the correlation between estimated mode shapes and target mode shapes), as well as the root mean square relative error (RMSRE), which, by taking into account all errors, represents a sort of "final score" that helps to measure the overall response misfit of each algorithm (refer to Paragraph 4 for more details about its calculation).

Judging by the best achieved cost function value and the RMSRE about the modal data, all algorithms are found to be able to accomplish at least decent minimization, except for GPS, which ends up with an objective value relatively far from zero, the known global minimum. Simulated annealing and GA perform similarly, returning decent values of RMSRE, which gives some hopes the algorithms were actually able to get quite close to the global optimum in the input space too. For what concerns these two optimization techniques, all MAC values are found to be very close to 1, suggesting very high correlation between computed mode shapes and target mode shapes, while natural frequencies seem to be the most discerning type of modal data in this case, as the results reveal a higher level of error (while still showing generally good agreement with target frequencies), particularly for the last three natural frequencies of the system.

Bayesian optimization really stands on its own, showing much better results in terms of best cost function value achieved and, consequently, agreement between target modal data and updated modal data. Indeed, accuracy levels regarding both mode shapes and natural frequencies (which, once again, take up most of the discrepancy between target and computed data) are quite remarkable, especially considering these results were obtained with only 100 iterations (a half of what allowed for the other algorithms).

Admittedly, SA and GA are optimization techniques that would require a higher number of iterations to perform at their best, but here the number of function evaluations was kept to a minimum to (1) cut short computational time, as many optimization runs involving a quite expensive objective function were executed, and (2) enable a comparison between these algorithms and Bayesian optimization, which number of iterations must instead kept small to reduce computational time due to modeling and point selection.

Mode n.	Freq.^{TARG} (Hz)	Generalized pattern search 200 fun. eval. Freq.^{UPD} (Hz)	Freq. err. (%)	MAC (-)	Simulated annealing (Strat. 2) 200 fun. eval. Freq.^{UPD} (Hz)	Freq. err. (%)	MAC (-)
1	4,39	4,38	338,22	1,0000	4,41	341,22	1,0000
2	6,50	6,53	226,48	0,9999	6,48	223,96	1,0000
3	22,70	22,66	655,49	0,9995	22,54	651,17	1,0000
4	29,15	29,84	645,89	0,9998	29,89	647,19	0,9999
5	34,57	38,89	677,76	0,9702	35,22	604,45	0,9999
6	38,75	40,22	570,37	0,9521	40,18	569,65	0,9961
RMSRE (%)			4,16			1,43	
Best cost f.			0,1359			0,0502	

Mode n.	Freq.^{TARG} (Hz)	Genetic Algorithm 200 fun. eval. Freq.^{UPD} (Hz)	Freq. err. (%)	MAC (-)	Bayesian optimization (LCB) 100 fun. eval. Freq.^{UPD} (Hz)	Freq. err. (%)	MAC (-)
1	4,39	4,40	340,31	1,0000	4,39	339,13	1,0000
2	6,50	6,54	226,80	1,0000	6,51	225,30	1,0000
3	22,70	22,74	658,06	0,9998	22,74	657,92	1,0000
4	29,15	29,52	637,96	1,0000	29,17	629,36	1,0000
5	34,57	36,80	636,09	0,9956	34,23	584,67	0,9999
6	38,75	38,95	549,13	0,9955	38,74	545,68	0,9999
RMSRE (%)			1,92			0,29	
Best cost f.			0,0511			0,0070	

Table 13. Optimization results in the output space, obtained with generalized pattern search (GPS), simulated annealing, Genetic Algorithm and Bayesian optimization. For each mode considered, the updated value and the target value are reported, as well as the relative error and the MAC value. Also, the RMSRE and the best achieved objective function value are reported for each algorithm.

Updating parameter	Target value	Generalized pattern search 200 fun. eval.		Simulated annealing (Strat. 2) 200 fun. eval.	
		Updated value	**Rel. error** (%)	**Updated value**	**Rel. error** (%)
E_A (GPa)	70,00	72,00	2,86	72,36	3,36
v (-)	0,32	0,40	25,00	0,32	-0,75
k_1 (kN/m)	1,00E+04	3,00E+04	200,00	2,47E+04	147,25
k_2 (kN/m)	2,00E+03	3,12E+03	56,00	2,08E+03	4,17
k_3 (kN/m)	2,00E+04	1,46E+04	-26,80	1,31E+04	-34,52
RMSRE (%)		94,33		67,68	

Updating parameter	Target value	Genetic Algorithm 200 fun. eval.		Bayesian optimization (LCB) 100 fun. eval.	
		Updated value	**Rel. error** (%)	**Updated value**	**Rel. error** (%)
E_A (GPa)	70,00	72,28	3,26	70,37	0,53
v (-)	0,32	0,37	15,00	0,34	4,81
k_1 (kN/m)	1,00E+04	8,11E+03	-18,90	9,47E+03	-5,32
k_2 (kN/m)	2,00E+03	2,44E+03	21,96	1,93E+03	-3,56
k_3 (kN/m)	2,00E+04	1,53E+04	-23,64	1,98E+04	-1,08
RMSRE (%)		18,08		3,62	

Table 14. Optimization results in the input space, obtained with generalized pattern search (GPS), simulated annealing, Genetic Algorithm and Bayesian optimization. For updating parameter, the updated value and the target value are reported, as well as the relative error between these two. Also, the RMSRE is computed for each algorithm, which represents a "final score" of the optimization results.

The results relative to the input space, and so the updated parameters (that generate the updated modal properties just discussed), are displayed in Table 14. The table reports, for each updating parameter, the target value, the updated value and the relative error obtained using the four optimization techniques. In general, the parameters' estimations show a much higher error when compared to the results obtained in the output space (i.e., updated modal data), denoting that the penalty function of this specific updating setup is scarcely sensitive to even significant changes of some parameters. To be more precise,

we can say that the dynamic response of the structure is almost independent on the springs that model the rigidity of the supports, as the majority of the errors are due to these parameters, while the estimation of the material Young's modulus obtained with all the algorithms are quite accurate. In light of the above, GPS, SA and GA attain good estimations of E_A, but fail at getting close to the correct values of k_1, k_2 and k_3, thus generating high errors. The only technique that achieved accurate results for what concerns the last three parameters is Bayesian optimization, in agreement with the results obtained in the output space. In fact, Bayesian optimization reached by far the best accuracy on all parameters, keeping the error levels to a minimum, with only half the objective function evaluations performed by the other algorithms (100 instead of 200). In summary, Bayesian optimization returns very encouraging results, significantly outperforming the other global optimization techniques considered.

The total optimization time employed by each algorithm is reported in Table 15, which also reports the total time required to attain the solution of the secondary optimization problems found in the Bayesian optimization method (modeling and point selection time).

	Generalized pattern search	**Simulated annealing**	**Genetic Algorithm**	**Bayesian Optimization**
Modeling and point selection time (s)	-	-	-	73
Total optimization time (s)	1358	1363	1378	756

Table 15. This table reports the elapsed optimization time for each technique. This computational times are obtained when using an average laptop computer. The modeling and point selaction time is relative to the computational cost of the secondary optimization problems occurring in Bayesian optimization. The rest of the algorithms, being much lighter then BO, employ an amount of time roughly eqaul to the computational time of the objective function times the number of total function evaluations.

As the time required to run the algorithm is minimal for GPS, SA and GA, the total optimization time practically equals the time used to compute the cost function times the number of evaluations. For what concerns Bayesian optimization, since the number of observations is relatively low (100 at the end of the process), the time employed to deal with the maximization of the marginal log-likelihood and the maximization of the

acquisition function (both operations requiring to evaluate thousands of predictions, each of them with a computational complexity of $\mathcal{O}(N^3)$,where N is the number of observations used to train the GP at each iteration) is not particularly impacting. As such, as only half of the function evaluations are used, Bayesian optimization enables to save a significant amount of computational time.

4.3 Case-study 3: structural identification of a historical building, the Mirandola bell tower

This case-study, which examines the bell tower of the Santa Maria Maggiore Cathedral in Mirandola (Italy), adds several layers of complexity to what already seen in the previous one. Moreover, it will serve both as a numerical case study, to further compare the performance of Bayesian optimization to the other algorithms, and as an experimental case study, allowing to finally test Bayesian optimization in a real-case model updating scenario. For this last application, model updating is used as a mean for structural damage assessment. Results stemming from this model updating procedure, carried out through the use of a Bayesian optimization approach, will be compared to the damage analysis of the tower bell structure conducted in the paper "Dynamic investigation on the Mirandola bell tower in post-earthquake scenarios" (Zanotti Fragonara, Boscato, & Ceravolo, 2017). As most aspects of this final application are based on the work done by the researchers, to clarify eventual dubious aspects of this dissertation and to gain further details about the subject, it is recommended to check out the cited paper. In fact, both the numerical and the experimental applications take advantage of the same finite element model, the same updating setup and the same experimental data of the damage study made in 2017.

Figure 39. Cathedral of Santa Maria Maggiore, Mirandola (Italy). The church was severely damaged by the seismic event of May 2012.

The numerical application of this model updating problem for damage assessment, which makes use of target input parameters and modal data in the same fashion of the previous case-studies, will be carried out in Paragraph 4.3.1, while the experimental case study, that will make use of the identified modal properties of the tower, will be presented in Paragraph 4.3.2.

As mentioned, the objective of the study was to assess the level of damage of a historical building by means of a model updating procedure. The bell tower of the Santa Maria Maggiore cathedral in Mirandola (Figure 39), built almost entirely in masonry, was struck by a seismic event that took place in Emilia (Italy) in May 2012, which caused quite extensive damage. The bell tower, which is square in plan with dimension of 5,90 m and has a height of 48 m, is represented in Figure 40. It is attached to the cathedral through the apse arches and the nave walls, and to the rectory building by means of the rectory wall. The post-earthquake damage of the tower is visible in Figure 41, where crack pattern and material leakage are highlighted.

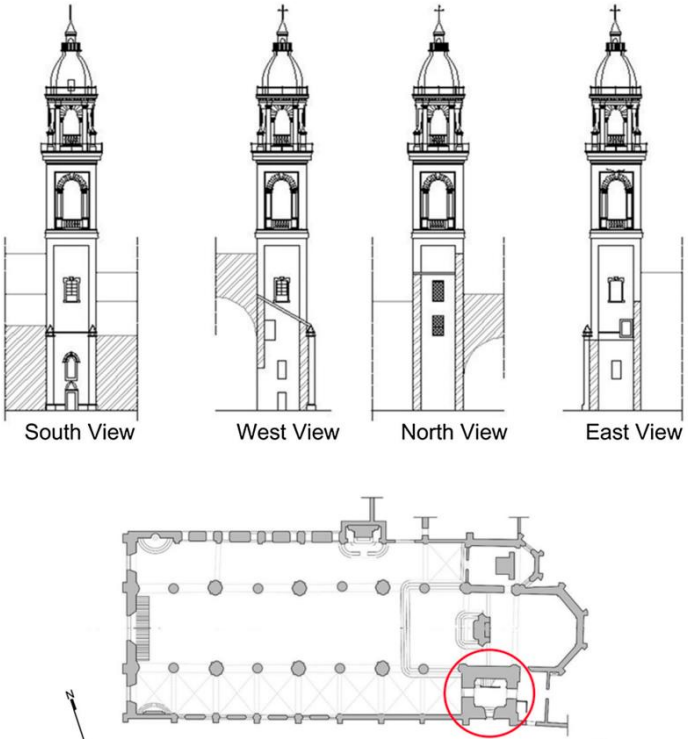


Figure 40. Main fronts and plan of the bell tower and the church. *(Zanotti Fragonara, Boscato, & Ceravolo, 2017)*.

The dynamic response of the bell tower was evaluated through operational modal analysis using ambient vibrations, a non-destructive procedure that enables to estimate the modal properties of the structure. To this extent, eight accelerometers were used: the instrumentation setup is visible in Figure 42. The system identification was carried out using the Stochastic Subspace Identification (SSI) algorithm, further details about pre-processing operations and the identification process can be found in (Ceravolo, Pistone, Fragonara, Massetto, & Abbiati, 2016). The same channels setup will be employed in the finite element model to compute modal data used for updating in both numerical and experimental analyses.

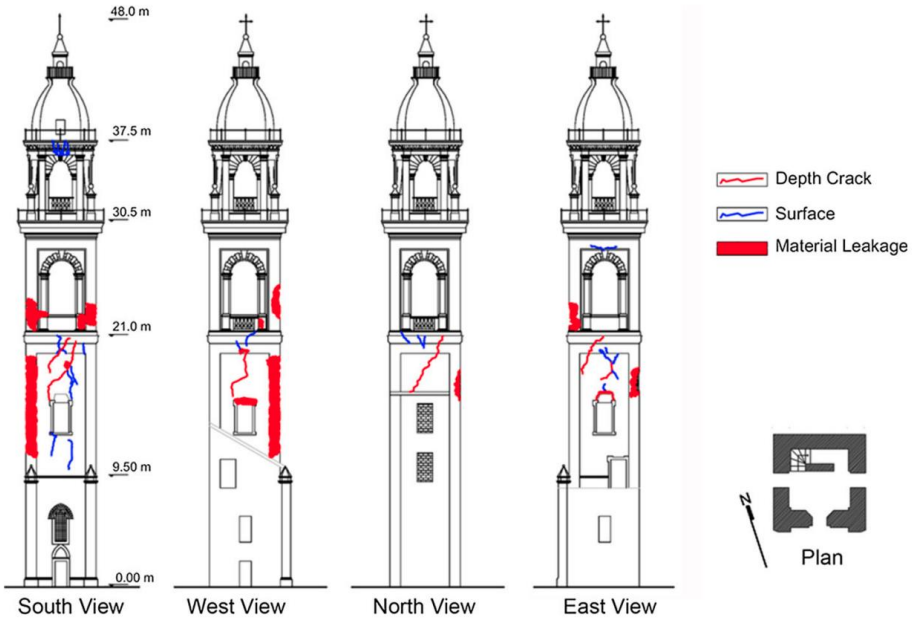
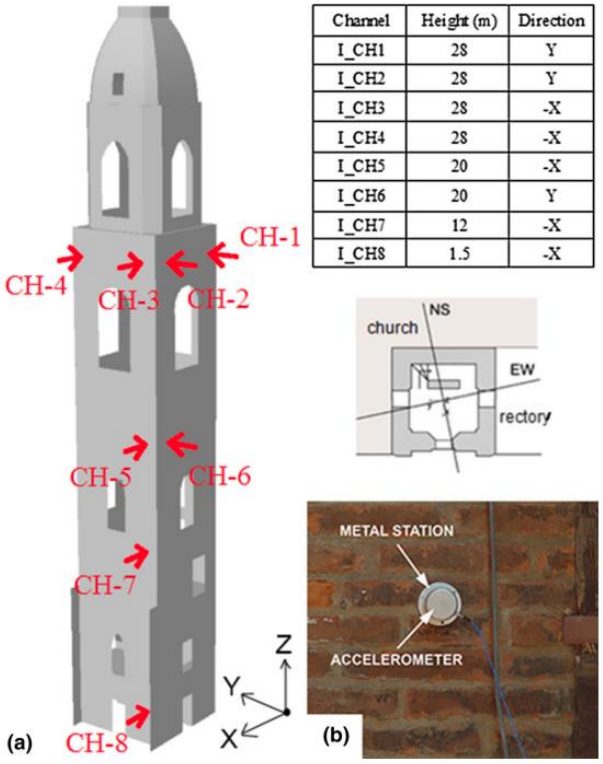


Figure 41. Post-earthquake damage of the bell tower. Crack patterns and material leakage are visible in red. *(Zanotti Fragonara, Boscato, & Ceravolo, 2017)*.

Use is made of the same finite element model of the original paper, built with the FEA software Ansys (Figure 43). The structure is divided in five sub-parts at different levels of the tower: all the elements belonging to one sub-part are connoted by the same material properties. As the nature of the interaction between the bell tower and the adjacent building is unknown, six linear springs were used in the FE model, located in correspondence of the architectonic constrains.



Channel	Height (m)	Direction
I_CH1	28	Y
I_CH2	28	Y
I_CH3	28	-X
I_CH4	28	-X
I_CH5	20	-X
I_CH6	20	Y
I_CH7	12	-X
I_CH8	1.5	-X

Figure 42. **a.** The instrumentation setup and orientation of the accelerometers employed. **b.** Detail of an accelerometer installed on the tower brick masonry walls. *(Zanotti Fragonara, Boscato, & Ceravolo, 2017)*.

Updating parameters.

Following the approach of the original paper, the elasticity modulus of each of the first four sub-parts (E_1, E_2, E_3, E_4), the Poisson's ratio ν and the stiffnesses of the six springs (k_1, k_2, k_3, k_4, k_5, k_6) were included in the updating procedure, for a total of 11 updating parameters. Estimating the elasticity modulus of the material through a model updating procedure enables to assess the level of damage of the structure, under the assumption that areas corresponding to low stiffness values denote high levels of structural damage. The elasticity modulus of the fifth sub-part was discarded, as the instrumentation placement couldn't capture any motion of the fifth level, disqualifying any attempt to estimate parameters relative to that part of the tower. The Poisson's ratio, which (marginally) affects the modal properties of the structure, is included in the updating procedure as its value is uncertain, being suggested only by agreed-upon literature and

normative values. The six linear springs that serve to model the degree of constrain provided by the walls and arches are obviously incorporated in the updating problem as their values are unknown. All updating parameters are also visible in Figure 43.

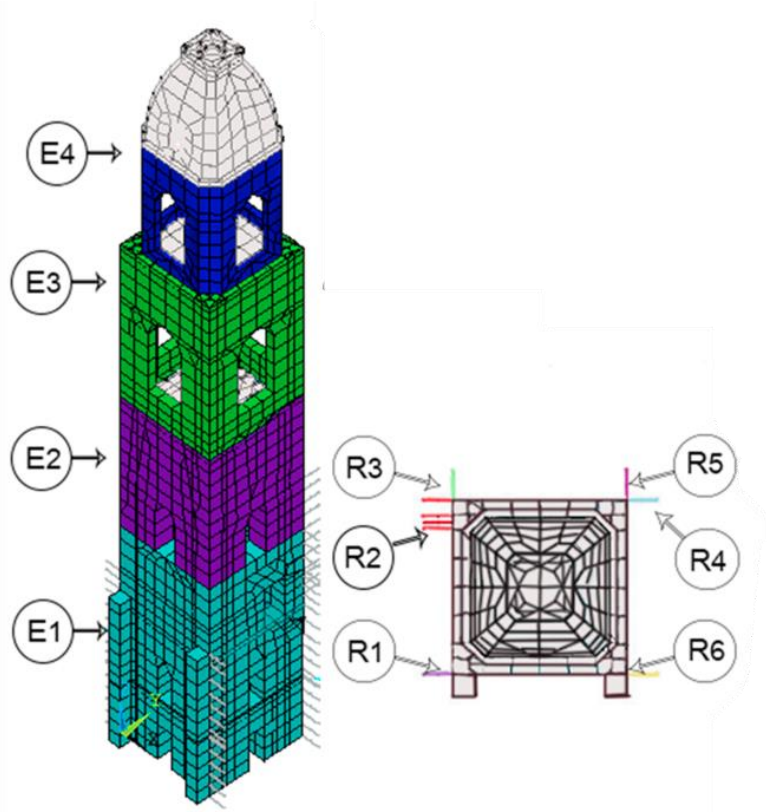


Figure 43. FE model discretization and sub-structure parametrization used for the updating procedure (left). Location of the springs used to model the interaction with the adjacent buildings (right). *(Zanotti Fragonara, Boscato, & Ceravolo, 2017)*.

Modal data used for updating.

The modal data that will be employed in the model updating procedure consists of the first six modes of the building, since these were found to be the most reliable identified modes in the original study. This is perfectly normal, as experimental data reliability typically decays as frequency increases. As such, the first six natural frequencies and the associated mode shapes will be used for both the numerical case study (where arbitrary target data will be used) and the experimental case study (that will make use of identified modal data).

The natural frequencies and the MAC values are arranged together to compute the misfit between the computed response and the target response. To this extent, the penalty function described in Paragraph 4 (Penalty function.) will be considered. As such, if the computed modal data and target (or identified) modal data are identical, the (always positive) penalty function P is perfectly minimized at 0, which constitutes the global optimum of the function, provided formal well-posedness of the updating problem (this eventuality is not possible when using experimental modal data, as the objective function global optimum is generally found at higher values than zero, see Paragraph 4.3.2).

Characteristics and complexities of the case-study.

The characteristics and the peculiarities of this case-study (particularly for what concerns the optimization problem), that are shared by both the numerical and experimental applications, are now briefly discussed, while case-specific aspects are further examined in the respective paragraphs. This case-study adds several layers of complexity to what already seen in case-study 2. In fact, besides the FE model expensiveness (which solution takes about 6 seconds with an average consumer laptop), the updating problem is characterized by (1) a very high number of dimensions, which enormously increases the computational complexity of the optimization problem, (2) a very wide optimization space, as very loose bounds will be placed on the spring parameters to reflect the high uncertainty we have on the level of constrain provided by the adjacent buildings, (3) a very variable level of sensitivity to the updating parameters, as these comprehend divers material properties and link-element features. All these elements together generate an extremely difficult optimization problem, featuring a particularly insidious and expensive objective function to minimize. The optimization problem is close at being ill-conditioned on the practical level, being the objective highly non-convex. All in all, this updating problem represents a very challenging playground to showcase the capabilities of Bayesian optimization.

4.3.1 Numerical case study

In this paragraph, the just described updating problem will be addressed at the numerical level to assess the validity of the updating technique used. Indeed, it makes no sense to deal with a real-case damage assessment problem by using a faulty methodology. This will be done employing arbitrary target modal data that will enable to firstly check the feasibility of the updating problem and its posedness, and secondly compare the performance of the four optimization techniques, likewise what already done with the previous case-studies.

4.3.1.1 Model updating setup

Target parameters and optimization boundaries.

In this setup, the parameters used for updating and the chosen target values are summarized in Table 16. The optimization space is delimited by a lower bound and an upper bound, defined for each updating variable, as found in the table.

Notice that the optimization range of the link-element parameters spans through several orders of magnitude, generating an extremely wide optimization space. The wide optimization range reflects the high uncertainty about the boundary conditions, which may contribute significantly (or, on the contrary, negligibly) to the dynamic response of the structure. The optimization bounds of the elasticity moduli take into account values suggested by literature and Italian regulations for brick masonry, while allowing to capture the level of damage suffered by the structure.

Updating parameter	Optimization bounds		Target value
	Lower bound	Upper bound	
E_1 (GPa)	0,375	6	0,5
E_2 (GPa)	0,375	6	1,5
E_3 (GPa)	0,375	6	1
E_4 (GPa)	0,375	6	4,5
k_1 (N/m)	1,00E+03	1,00E+10	4,10E+06
k_2 (N/m)	1,00E+03	1,00E+10	2,00E+03
k_3 (N/m)	1,00E+03	1,00E+10	1,60E+06
k_4 (N/m)	1,00E+03	1,00E+10	6,30E+05
k_5 (N/m)	1,00E+03	1,00E+10	1,50E+09
k_6 (N/m)	1,00E+03	1,00E+10	1,00E+05
ν (-)	0,40	0,50	0,45

Table 16. Parameters selected for updating and associated optimization bounds.

Target modal data.

As mentioned, the first six modes of the bell tower are used in this model updating setup. Table 17 shows the target modal data, in particular the first six natural frequencies (mode shapes are not reported here for practical reasons). The target modal data is generated by the (arbitrarily chosen) target parameters already shown in Table 16.

Mode shapes are measured at the location of the sensors, as described in Figure 42. Vibration data is gathered at the corresponding FE model nodes when performing the updating procedure to maintain coherence between computed mode shapes and experimental mode shapes.

Mode n.	Freq.^{TARG} (Hz)
1	0,72
2	0,74
3	1,96
4	2,77
5	2,96
6	4,57

Table 17. Target natural frequencies (generated by the set of target parameters).

4.3.1.2 Bayesian optimization results and performance comparison of the algorithms

In the following paragraph, the results of the model updating procedure obtained through the use of each optimization algorithm are presented, and the performances of each algorithm are compared: particular focus will be placed on the number of objective function evaluations performed relatively to the achieved accuracy, both regarding the output data (modal properties) and the input data (updating parameters), and on the total computational time.

In this case study, generalized pattern search, simulated annealing and Genetic Algorithm will be allowed to perform a maximum of 1000 evaluations of the objective function. Bayesian optimization, as it features much better sampling efficiency, is allowed to evaluate the objective function only 500 times. To deal with the variability of results, simulated annealing and GA are now run 5 times; the average results or a representative run out of the 5 executions is further on considered.

In this paragraph the Bayesian approach is employed to minimize the cost function of the current updating problem. After computing the initial seed of points, the effects of transforming the input parameters as well as the impact of using four different kernel functions, namely the exponential kernel, the Matérn 3/2 kernel, the Matérn 5/2 kernel and the squared exponential kernel, will be explored by evaluating the validity of the

Gaussian Process through cross-validation tests. Finally, Bayesian optimization is run using the lower confidence bound (LCB) acquisition function and the results will be discussed. Following the usual rule of dumb, in this specific case study, where 11 parameters are being updated, the initial seed size is set to 220 points.

Transformation of optimization variables.

A logarithmic transformation is applied to the input parameters, as it was found to enhance the GP regression quality in the previous case-studies. Since in this case the seed size is large, the validity of the surrogate is assessed via a non-exhaustive cross-validation technique, namely a $k-$fold loss techinique, after fitting the Gaussian Process to the initial seed points (at this stage the ARD Matérn 5/2 kernel was used).

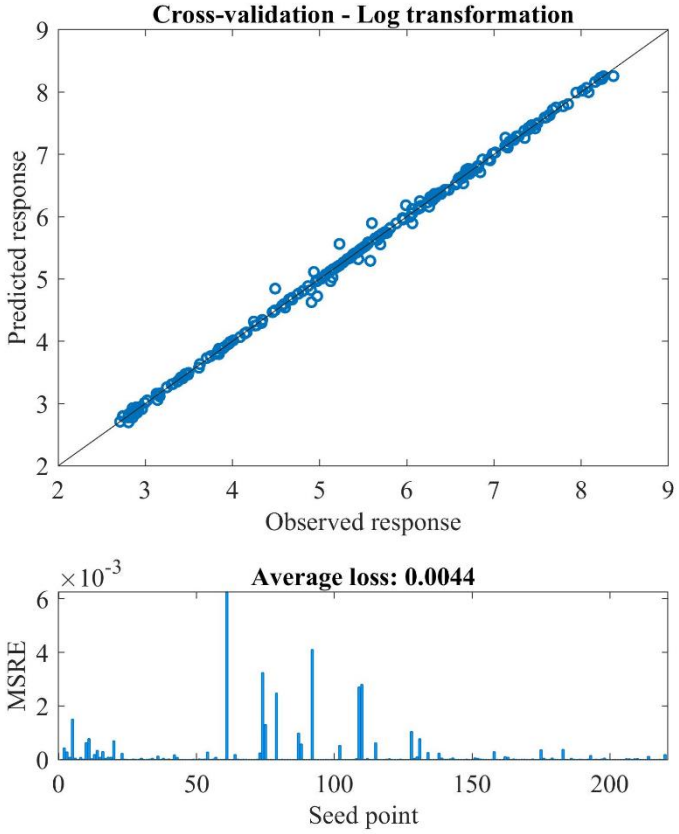


Figure 44. 10-folds CV results of a GP fitted using log-transformed variables. The plot at the top shows the predicted response against the observed response: the data should lay as close as possible to the 45˚-sloped line. On the bottom, the diagram displays the mean squared relative error (MSRE) between the objective values predicted by the GP fitted using the training dataset and the corresponding left-out observations.

Figure 44 shows the results of the $10-$folds cross-validation test. The diagram on the top shows the predicted response against the observed response: the data should lie as close as possible to the 45° sloped line for a good model validation. On the bottom, the diagram displays the square relative error between the objective values predicted by the GP fitted on the training dataset and the corresponding left-out observations. The model is seen to validate very well, as the cross-validation test returns very average low modeling loss.

Kernel functions and parameters' length scales.

Once again, the choice of the kernel function to be used is driven by a cross-validation test, since some kernels may happen to be more suitable at modeling the underlying objective function specific to this updating problem, resulting in surrogate models with enhanced validity.

	ARD exponential kernel	ARD Matérn 3/2	ARD Matérn 5/2	ARD squared exponential kernel
MSE (-)	0,0108	0,0081	0,0044	0,0101

Table 18. Cross validation loss when using different kernel function. The mean square error (MSE) is reported for each kernel, averaging the validation losses obtained using the 10-fold CV.

Table 18 contains the mean square error given by a 10-fold cross-validation test of four Gaussian Processes, fitted using the ARD exponential, the ARD Matérn 3/2, the ARD Matérn 5/2 and the ARD squared exponential kernels. Generally, all surrogate models are seen to validate quite well. In this case, the ARD Matérn 5/2 kernel is found to be the most suitable at modeling the objective function, returning excellent validation results. Anyhow, all kernels are found to be sufficiently suitable for the modeled objective function.

Parameter	Length scale (-)
E_1	3,59
E_2	3,58
E_3	5,76
E_4	456953,46
k_1	61,91
k_2	216,27
k_3	45,47
k_4	27,64
k_5	435543,59
k_6	331617,01
ν	48,55

Table 19. Parameters' length scales obtained by maximizing the marginal log-likelihood. The higher sensitivity is found for the elasticity moduli of the bell tower walls (except for the fourth one). The length scale of elasticity modulus of the fourth sub-part is very high, suggesting the updating problem is not sensitive enough to this parameter.

Since automatic relevance determination kernel functions are employed, it is possible to extract the length scale of each updating parameter from the hyperparameters of the Gaussian Process fitted to the initial seed of points. The length scales (reported in Table 19) provide information about the sensitivity of the problem to the updating variables, recalling that a high length scale suggests low sensitivity, while a low length scale indicates high sensitivity. As expected, also in this case the eleven length scales are not of the same order of magnitude. In fact, the parameters which mostly affects the problem (i.e., the system modal response are the elasticity moduli: such behavior is reasonable, as material rigidity is usually blamed to significantly impact modal properties. The elasticity modulus of the fourth sub-part has by far the least impact on the updating problem, this is due to the scarce presence of sensors at that level of the building, which disqualifies from capturing the necessary vibration information. Therefore, we should not expect very reliable estimations of E_4, neither in the numerical case, nor in the experimental one. The problem is not very sensitive to changes of the Poisson's ratio ν either: this makes perfect

sense as damping has only marginal effect on modal properties. For what concerns the springs, these are generally found to have lesser effects on the modal response, compared to the elasticity moduli of the first three sub-structures. In particular, k_5 and k_6 are found to have the lower impact on the system modal response, while k_1, k_3 and k_4 feature higher sensitivity. Notice that the length scales, computed by maximizing the likelihood (see Paragraph 2.3.2), are relative to the 220 randomly sampled points: since the problem sensitivity to the updating parameters varies across the input space, different length scales may be obtained when changing the location of the initial seed points (this could be particularly true for the springs, which optimization range is extremely wide).

Optimization procedure.

After choosing to log transform the input variables and selecting the ARD Matérn 5/2 kernel, the Bayesian optimization process is finally carried out using lower confidence bound (LCB) acquisition function, which was often found to be the most effective in the previous case-studies, providing a reasonable balance between exploitation and exploration. Throughout the optimization process, Figure 45 (at the top) displays the objective value at the randomly sampled seed points in red and the objective value at the sampling point chosen by the algorithm at each iteration in blue.

It can be noticed how the first selected sampling point (blue color) already represents a massive improvement over the best cost function values found when computing the seed points, denoting that the Gaussian Process is able to model the objective function impressively well. As the GP is updated with newly sampled points, LCB steadily converges towards a minimum, gradually improving the accuracy of the optimization solution. For additional clarity, the best computed objective against the iteration number is shown at the bottom of Figure 45. The best obtained objective is 0,07424, which is fairly close to zero, the (known) global optimum value in the output space. Also, it is possible to see that Bayesian optimization struggles at further improving the result after

350 iterations, suggesting that the algorithm could be trapped in a particularly insidious local minimum, which happens to be very close to the global one (in the output space).

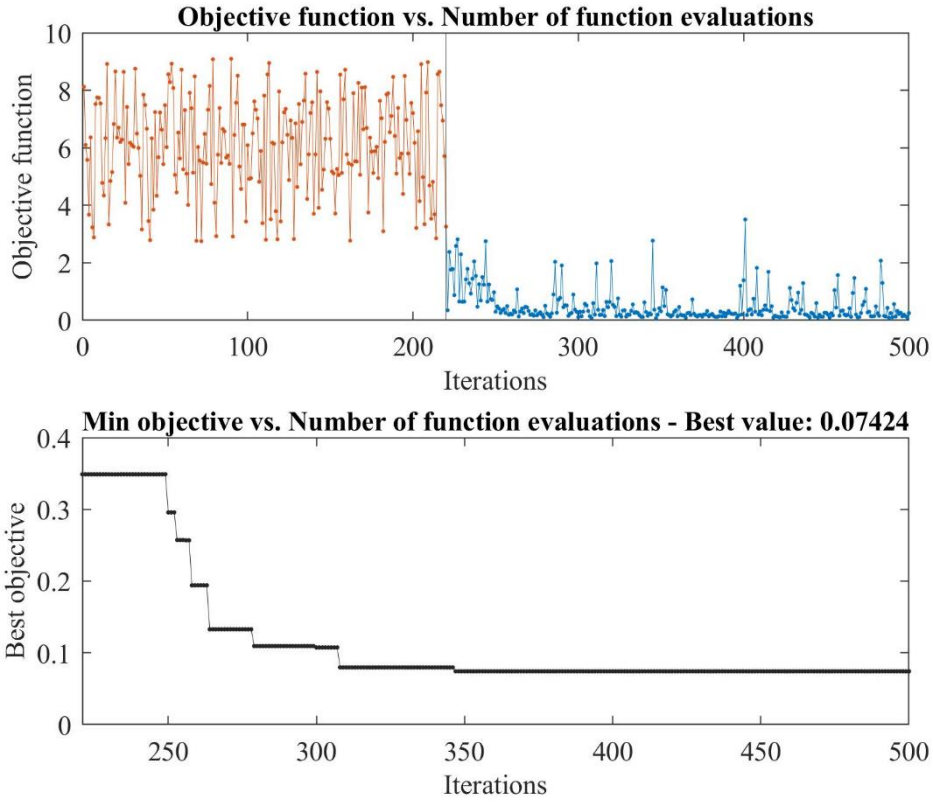


Figure 45. At the top, it is possible to visualize the behavior of the Bayesian optimization throughout the optimization process. The objective value at the randomly sampled seed points is displayed in red, while the objective at points selected by LCB is displayed in blue. At the bottom, the beast objective function obtained during the obtained optimization process is displayed. Notice how BO struggles at further improving the best achieved objective after 350 function evaluations.

Optimization time.

For what concerns the optimization time, since the number of total observations is quite high, the time required to run the algorithm (modeling and point selection time) substantially impacts the total optimization time, which depends only in part on the time required to evaluate the objective function. This is visible in Figure 46, which shows the total optimization time against the number of function evaluations, as the sum of the objective evaluation time and the modeling and point selection time (which is required to maximize the log likelihood and to maximize the acquisition function). Still, enabling to

minimize the objective function reasonably well in 500 iterations only (actually, in this case, about 350 would be sufficient to generate the same optimization results), Bayesian optimization demonstrates to be convenient in terms of required computational time once again.

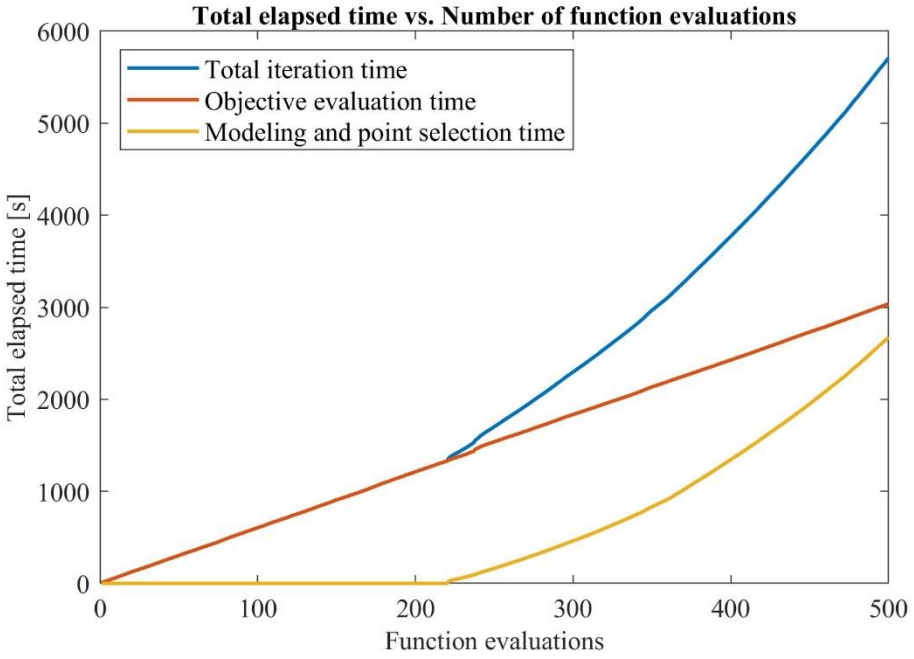


Figure 46. The figure displays the total optimization time as the sum of the objective evaluation time (the cumulative time employed for evaluating the objective function) and the modeling and point selection time (the cumulative time employed for solving the secondary optimization problems, i.e., the maximization of the marginal log-likelihood and the maximization of the acquisition function). As computing a prediction has a computational complexity of $\mathcal{O}(N^3)$, the modeling and point selection time gradually increases as the number of observations N grows.

Performance comparison of the optimization techniques.

Finally, the comparison of the results obtained from the four optimization techniques enables to evaluate the performance of Bayesian optimization and to assess the computational feasibility of the updating problem in question. The results relative to the output, that are the raw (best) cost function value and the relative estimated modal data, are shown in Table 20, for each technique. The table reports the relative errors between the updated natural frequencies and the known target values (generated by the set of target parameters), the MAC values (that measure the correlation between estimated mode

shapes and target mode shapes), as well as the root mean square relative error (RMSRE), which, by taking into account all errors, represents a sort of "final score" that helps to measure the overall response misfit of each algorithm (see Paragraph 4 for more details).

		Generalized pattern search 1000 fun. eval.			**Simulated annealing (Strat. 2)** 1000 fun. eval.		
Mode n.	**Freq.^{TARG}** (Hz)	**Freq.^{UPD}** (Hz)	**Freq. err.** (%)	**MAC** (-)	**Freq.^{UPD}** (Hz)	**Freq. err.** (%)	**MAC** (-)
1	0,72	0,74	3,17	0,86	0,70	-3,06	0,25
2	0,74	0,90	21,26	0,84	1,01	35,63	0,20
3	1,96	2,09	6,36	0,98	1,89	-3,54	0,99
4	2,77	2,47	-10,58	0,93	2,52	-8,75	0,02
5	2,96	2,96	0,26	0,35	3,01	1,76	0,41
6	4,57	4,24	-7,19	0,70	3,84	-15,86	0,23
RMSRE (%)			22,89			52,25	
Best cost f.			0,9472			2,4817	

		Genetic Algorithm 1000 fun. eval.			**Bayesian optimization (LCB)** 500 fun. eval.		
Mode n.	**Freq.^{TARG}** (Hz)	**Freq.^{UPD}** (Hz)	**Freq. err.** (%)	**MAC** (-)	**Freq.^{UPD}** (Hz)	**Freq. err.** (%)	**MAC** (-)
1	0,72	0,56	-22,35	0,86	0,73	0,87	0,99
2	0,74	0,75	0,23	0,82	0,78	4,28	0,99
3	1,96	1,90	-2,99	0,98	1,96	-0,41	1,00
4	2,77	2,24	-18,92	0,99	2,82	1,89	0,99
5	2,96	2,90	-2,03	0,27	2,98	0,68	0,98
6	4,57	3,92	-14,15	0,80	4,62	1,26	0,99
RMSRE (%)			29,69			1,63	
Best cost f.			0,9436			0,0742	

Table 20. Optimization results in the output space, obtained with generalized pattern search (GPS), simulated annealing, Genetic Algorithm and Bayesian optimization. For each mode, the updated value and the target value are reported, as well as the relative error and the MAC value. Also, the RMSRE and the best achieved objective function value are reported for each algorithm.

Bayesian optimization, on the contrary, with only 500 objective evaluations, achieves quite impressive results when compared to the other techniques. The error about the frequencies is kept to a minimum (here, only the second mode is showing a slightly higher error), and so is the error about the mode shapes.

The results relative to the input space, and so the updated parameters (that generate the updated modal properties just discussed), are displayed in Table 21. The table reports, for each updating parameter, the optimization bounds, the target value, the updated value and the relative error obtained using the four optimization techniques. Among all parameters, the most interesting surely are the elasticity moduli of the four sub-structures, as these parameters respond to the main task of the updating problem, which is assessing the level of damage of the structure. The stiffnesses of the six springs are introduced in the updating procedure only because we are unaware of the degree of support provided by the adjacent architectonical elements and their impact on the dynamic response of the building (specifically, its modal properties). Moreover, given their extremely wide optimization range, identifying the right order of magnitude can be considered as sufficient. In light of the above, GPS, SA and GA show quite poor results, failing at attaining a good estimation of the first four parameters (and providing even worse estimations for the rest of the parameters). Bayesian optimization, instead, returns acceptable errors over the estimated values, showing good agreement with the first four in particular. Even if an error about the elasticity moduli about 25% may seem like a lot compared to what obtained in the previous case-studies, considering the extreme complexity of the optimization problem in question, these results are truly remarkable. Such a degree of accuracy already enables understanding which parts of the structure underwent the most damage, and which ones may be structurally healthy. For what concerns the springs, the order of magnitude is mostly recognized, except for three cases that really stand out: k_1, k_4 and k_6 estimations are quite off, suggesting that the algorithm was indeed stuck in a local minimum, as already foreseen looking at its behavior throughout the optimization process. Had Bayesian optimization been able to properly sort k_1, k_4 and k_6, it would have most probably returned better accuracy about the parameters of interest (i.e., E_1, E_2, E_3 and E_4).

Updating parameter	Target value	Generalized pattern search 1000 fun. eval.		Simulated annealing (Strat. 2) 1000 fun. eval.	
		Updated value	**Rel. error** (%)	**Updated value**	**Rel. error** (%)
E_1 (GPa)	**5,00E+08**	**8,25E+08**	**65,00**	**3,76E+08**	**-24,87**
E_2 (GPa)	**1,50E+09**	**8,25E+08**	**-45,00**	**5,77E+08**	**-61,52**
E_3 (GPa)	**1,00E+09**	**2,63E+09**	**162,50**	**5,54E+09**	**453,52**
E_4 (GPa)	**4,50E+09**	**1,50E+09**	**-66,67**	**5,79E+09**	**28,64**
k_1 (N/m)	4,10E+06	4,00E+08	9656,10	1,00E+10	243775,43
k_2 (N/m)	2,00E+03	4,00E+08	199999,00	8,45E+08	422499,57
k_3 (N/m)	1,60E+06	1,00E+03	-99,94	2,07E+06	29,29
k_4 (N/m)	6,30E+05	1,00E+03	-99,84	9,98E+09	1584451,36
k_5 (N/m)	1,50E+09	1,20E+09	-20,00	9,98E+09	565,41
k_6 (N/m)	1,00E+05	1,00E+10	9999900,00	3,41E+08	340644,44
v (-)	6,00E+04	6,00E+04	-0,02	6,04E+04	0,75
RMSRE (%)		96,31		229,62	

Updating parameter	Target value	Genetic Algorithm 1000 fun. eval.		Bayesian optimization (LCB) 500 fun. eval.	
		Updated value	**Rel. error** (%)	**Updated value**	**Rel. error** (%)
E_1 (GPa)	**5,00E+08**	**3,73E+08**	**-25,40**	**6,28E+08**	**25,66**
E_2 (GPa)	**1,50E+09**	**8,51E+08**	**-43,27**	**1,53E+09**	**1,86**
E_3 (GPa)	**1,00E+09**	**2,01E+09**	**101,00**	**8,19E+08**	**-18,13**
E_4 (GPa)	**4,50E+09**	**3,75E+08**	**-91,67**	**3,97E+09**	**-11,68**
k_1 (N/m)	4,10E+06	9,21E+09	224534,15	1,89E+04	-99,54
k_2 (N/m)	2,00E+03	1,00E+03	-50,00	5,62E+06	280990,00
k_3 (N/m)	1,60E+06	1,00E+03	-99,94	1,48E+06	-7,62
k_4 (N/m)	6,30E+05	1,00E+03	-99,84	1,01E+04	-98,40
k_5 (N/m)	1,50E+09	5,53E+09	268,67	1,75E+09	16,33
k_6 (N/m)	1,00E+05	1,00E+03	-99,00	8,18E+07	81733,90
v (-)	0,45	4,30E-01	-4,44	4,65E-01	3,31
RMSRE (%)		72,67		16,79	

Table 21. Optimization results in the input space, obtained with GPS, simulated annealing, Genetic Algorithm and Bayesian optimization. The updated value, the target value, the relative error (for each updating parameter) and the RMSRE (about the four elasticity moduli) value are reported.

In the damage assessment study of 2017, the updating procedure was carried out in batches: first, the springs were calibrated while holding the elasticity moduli constant, and only at a second stage, E_1, E_2, E_3 and E_4 were optimized using the link-element stiffness values previously estimated. As traditional optimization techniques were employed, this approach aimed at facilitating the optimization procedure by lowering the number of dimensions of the problem. Given the outcome of this numerical test, when using Bayesian optimization this approach can be avoided, as this technique is powerful enough to allow considering all parameters at once, cutting off the computational time and enhancing chances of individuating the real global function minimum.

	Generalized pattern search	Simulated annealing	Genetic Algorithm	Bayesian Optimization
Modeling and point selection time (s)	-	-	-	2672
Total optimization time (s)	6118	6213	6237	5708

Table 22. This table reports the elapsed optimization time for each technique, obtained using an average laptop computer. The modeling and point selaction time is relative to the computational cost of the secondary optimization problems occurring in Bayesian optimization. The rest of the algorithms, being much lighter then BO, employ an amount of time roughly eqaul to the computational time of the objective function times the number of total function evaluations.

The total required optimization time employed by each algorithm is reported in Table 22. As the time required to run the algorithm is minimal for GPS, SA and GA, the total optimization time practically equates the time required to compute the cost function times the number of evaluations. Bayesian optimization still allows to save some time to complete the optimization procedure. Nonetheless, as the number of observations is relatively high, in this occasion the secondary optimization problems (i.e., the maximization of the marginal log-likelihood and the maximization of the acquisition function) are relatively burdensome tasks, such to significantly extend the total optimization time.

4.3.2 Experimental case study

This case study, where experimental data is used, is fundamentally different from the previous ones. When dealing with identified modal properties, two major sources of error arise in model updating problems, which are not strictly related to the optimization problem (as discussed in Paragraph 2.1). The first one is due to the inherent limitations of the FE model, which may be source of huge errors by not being able to fully render the dynamic response of the real structure. The second source of error stems from the reliability of the identified natural frequencies and mode shapes. As finite element model updating is an inverse problem, extremely sensitive to even small errors, the cost function is not going to be minimized at zero when using real data, since significant misfit will endure between experimental and computed modal properties. Not only: due to the high liability of iterative model updating methods to small errors, the set of parameters corresponding to the penalty function global minimum may not be the better at representing the actual properties of the physical system. In other words, a local minimum could be a better solution of a damage assessment procedure or model calibration problem.

Anyhow, attaining a good solution of such a model updating problem still provides useful information for assessing the level of damage of the structure in analysis. In this paragraph, Bayesian optimization will be employed to this extent, and the obtained results will be qualitatively compared to the ones attained in the original study.

4.3.2.1 Model updating setup

Optimization boundaries.

In this setup, the optimization boundaries of the updating parameters are chosen following the model updating setup developed in the 2017 study. As a starting value of the elasticity modulus of the masonry walls, 1.5 GPa were considered, while for the constraint springs the maximum-rigidity condition was set to 100 GN/m. Table 23 shows the updating parameters and the related chosen boundaries. Note that in this case it was decided to

consider slightly tighter bounds for the link-element parameters compared to the original paper; also, the elasticity moduli lower bounds are looser compared to the respective upper bounds, to allow capturing even high levels of structural damage.

Updating parameter	Optimization bounds	
	Lower bound	Upper bound
$\mathbf{E_1}$ (GPa)	0,35	2
$\mathbf{E_2}$ (GPa)	0,35	2
$\mathbf{E_3}$ (GPa)	0,35	2
$\mathbf{E_4}$ (GPa)	0,35	2
$\mathbf{k_1}$ (N/m)	1,00E+03	1,00E+10
$\mathbf{k_2}$ (N/m)	1,00E+03	1,00E+10
$\mathbf{k_3}$ (N/m)	1,00E+03	1,00E+10
$\mathbf{k_4}$ (N/m)	1,00E+03	1,00E+10
$\mathbf{k_5}$ (N/m)	1,00E+03	1,00E+10
$\mathbf{k_6}$ (N/m)	1,00E+03	1,00E+10
v (-)	0,40	0,50

Table 23. Parameters selected for updating and associated optimization bounds.

Experimental modal data.

As mentioned, the first six identified modes of the structure are used in this model updating setup, as these were thought to be the most reliable ones. For details about the modal identification process, refer to the paper (Zanotti Fragonara, Boscato, & Ceravolo, 2017). According to their study, Table 24 shows the identified modal data that will be employed, namely the six natural frequencies and the associated six mode shapes.

The mode shapes are measured at the location of the sensors, as described in Figure 42. Vibration data is gathered at the corresponding FE model nodes when performing the updating procedure.

Mode n.	Freq.^{ID} (Hz)
1	0.720
2	0.743
3	1.963
4	2.766
5	2.955
6	4.567

Table 24. Identified natural frequencies of the bell tower that are used in the updating procedure.

4.3.2.2 Optimization using the Bayesian approach and assessment of results

In the following section, the Bayesian optimization approach will be used to assess the level of damage of the bell tower. After computing the initial seed of points and choosing the most suitable kernel function through model validation, Bayesian optimization is run using the lower confidence bound (LCB) acquisition function and the obtained results will be discussed.

Following the usual rule of dumb, in this specific case study, where 11 parameters are being updated, the initial seed size is set to 250 points.

Transformation of optimization variables.

A logarithmic transformation is applied to the input parameters, as it was found to be beneficial in the previous case-studies. Since in this case the seed size is large, the validity of the surrogate is assessed via a non-exhaustive cross-validation technique, namely a k −fold loss techinique, after fitting the Gaussian Process to the initial seed points (at this stage the ARD Matérn 5/2 kernel was used).

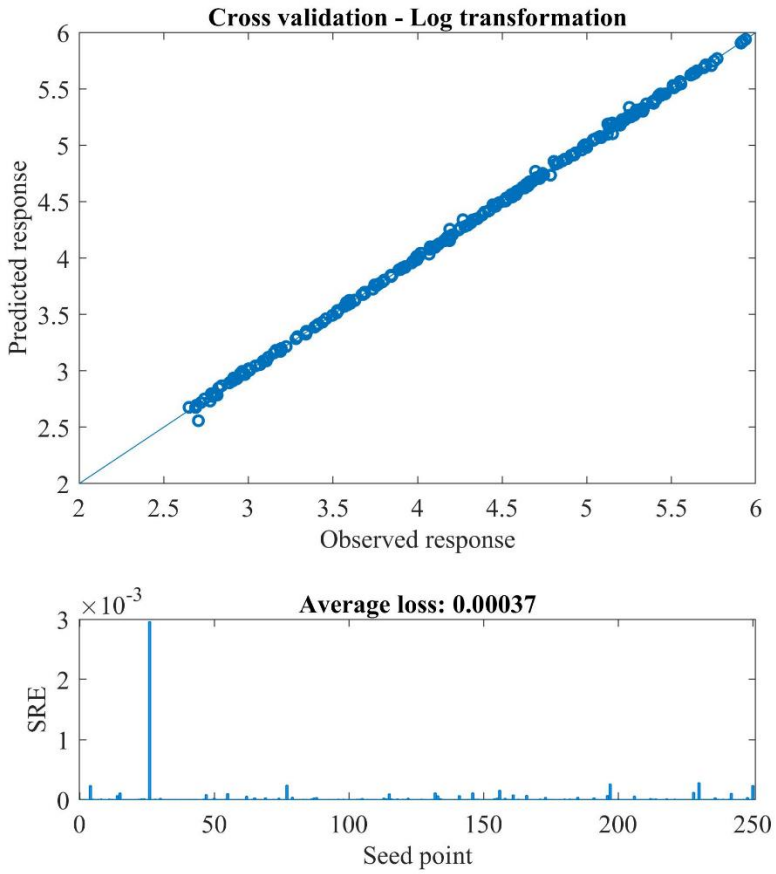


Figure 47. 10-folds CV results of a GP fitted using log-transformed variables. The plot at the top shows the predicted response against the observed response: the data should lay as close as possible to the 45˚-sloped line. On the bottom, the diagram displays the mean squared relative error (MSRE) between the objective values predicted by the GP fitted using the training dataset and the corresponding left-out observations.

Figure 47 shows the results of the 10 −folds cross-validation test. The diagram at the top shows the predicted response against the observed response: for a good model validation, the data should lay as close as possible to the 45˚ sloped line. At the bottom, the diagram displays the mean square relative error between the objective values predicted by the GP fitted on the training dataset and the corresponding left-out observations. Also in this case, the model is seen to validate remarkably well, as the cross-validation test returns a very low average loss.

Kernel functions and parameters' length scales.

Once again, the choice of the kernel function to be used is driven by a cross-validation test, since some kernels may happen to be more suitable at modeling the underlying objective function specific to this updating problem, resulting in surrogate models with enhanced validity.

	ARD exponential kernel	ARD Matérn 3/2	ARD Matérn 5/2	ARD squared exponential kernel
MSE (-)	0.00102	0,00056	0,00037	0,00046

Table 25. Cross validation loss when using different kernel functions. The mean square error (MSE) is reported for each kernel, averaging the validation losses obtained using the 10-fold CV.

Table 25 contains the mean square error given by the 10-fold cross-validation test of four Gaussian Processes, fitted using the ARD exponential, the ARD Matérn 3/2, the ARD Matérn 5/2 and the ARD squared exponential kernels. Generally, all surrogate models are seen to validate very well. Once again, the ARD Matérn 5/2 kernel is found to be the most suitable at modeling the underlying objective function, returning excellent validation results.

Since automatic relevance determination kernel functions are employed, it is possible to retrieve the length scale of each updating parameter from the hyperparameters of the Gaussian Process fitted to the initial seed of points. The length scales (reported in Table 26) provide information about the sensitivity of the problem to the updating variables, recalling that a high length scale suggests low sensitivity, while a low length scale indicates high sensitivity. The considerations made in the numerical case hold here as well: the parameters which mostly affects the updating problem are the elasticity moduli, exception made for the one of the fourth sub-part, as the scarce presence of sensors at that level of the building disqualifies from capturing the necessary vibration information. Therefore, we should not expect reliable estimations of E_4. The same considerations made for the numerical case about the Poisson's ratio and the stiffness parameters of the springs apply here as well.

Parameter	Length scale (-)
E_1	2,69
E_2	1,96
E_3	16,19
E_4	174122,97
k_1	76,69
k_2	1436,38
k_3	40,14
k_4	33,44
k_5	372727,04
k_6	1255994,23
ν	23,31

Table 26. Parameters' length scales obtained by maximizing the marginal log-likelihood. The higher sensitivity is found for the elasticity moduli of the bell tower walls (except for the fourth one). The length scale of elasticity modulus of the fourth sub-part is very high, suggesting the updating problem is not sensitive enough to this parameter. Thus, it is not possible to estimate the value of this parameter while using this specific updating setup.

Optimization procedure.

After choosing to log-transform the input variables and selecting the ARD Matérn 5/2 kernel, just like in the numerical case, the Bayesian optimization process is finally carried out using lower confidence bound (LCB) acquisition function, which provides good balance between exploitation and exploration. Figure 48 (on the top) displays the objective value at the randomly sampled seed points in red, and the objective value at the sampling points chosen by the algorithm at each iteration in blue.

Also in this experimental case, Bayesian optimization steadily converges towards a minimum, gradually improving the accuracy of the optimization solution. In this case study, since experimental data is used, the optimizer cannot be expected to converge at zero. In fact, as already mentioned, FE modeling deficiencies coupled with identified modal data inaccuracies lead to ineluctable misfit between computed and measured modal

response. For additional clarity, the best computed objective against the number of iterations is shown at the bottom of Figure 48. The objective value associated to this function minimum is 0,8343.

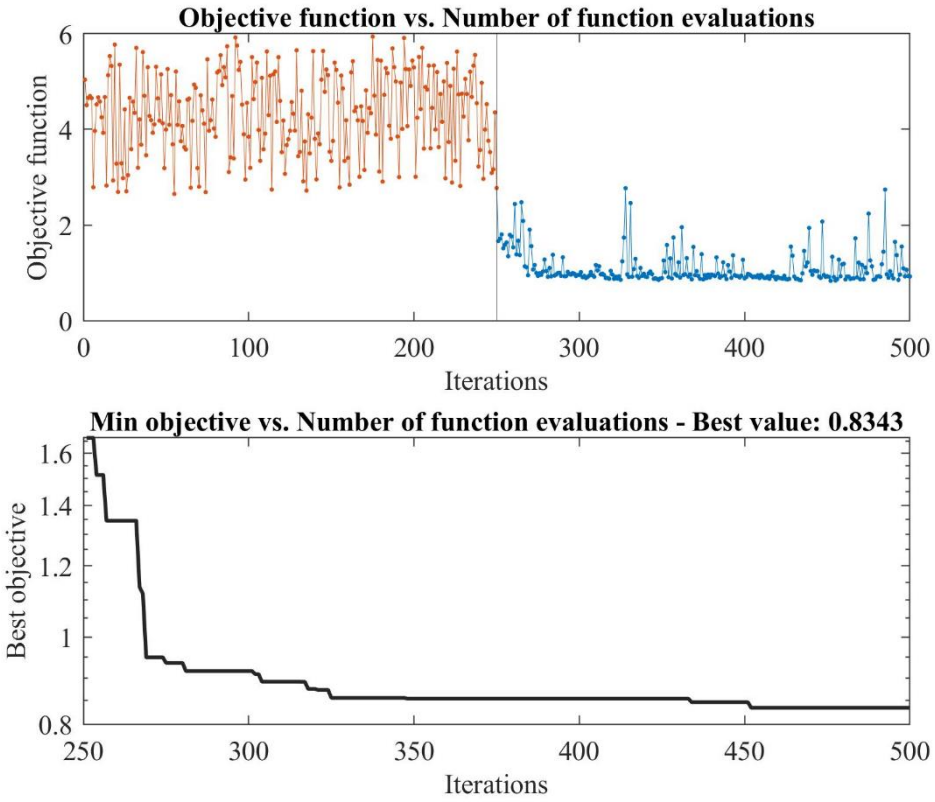


Figure 48. At the top, it is possible to visualize the behavior of the Bayesian optimization throughout the optimization process. The objective value at the randomly sampled seed points is displayed in red, while the objective at points selected by LCB is displayed in blue. At the bottom, the beast objective function obtained during the obtained optimization process is displayed. Notice how the objective function minimum is not close to zero in this case since experimental modal data is being used.

Optimization performance and considerations about results.

The results relative to the output, that are the raw (best) cost function value and the relative estimated modal data, are shown in Table 27, together with the modal parameters attained in the original paper as reference. The table reports the relative errors between the updated natural frequencies and the identified values (as reported in the previous paragraph), the MAC values (that measure the correlation between estimated mode

shapes and identified mode shapes), as well as the root mean square relative error (RMSRE), which, by taking into account all errors, represents a sort of "final score" that helps at measuring the overall response misfit of the algorithm (refer to Paragraph 4 for more details about its calculation). Finally, the updated modal parameters obtained through the updating procedure executed in the original study are reported as reference.

		Bayesian optimization (LCB) 500 fun. eval.			**Original paper** Ref. (2017)		
Mode n.	**Freq.^{ID}** (Hz)	**Freq.^{UPD}** (Hz)	**Freq. err.** (%)	**MAC** (-)	**Freq.^{UPD}** (Hz)	**Freq. err.** (%)	**MAC** (-)
1	0,68	0,67	1,75	0,98	0,68	0,20	0,98
2	0,72	0,71	1,17	0,96	0,69	3,80	0,98
3	1,41	1,64	16,22	0,87	1,73	18,30	0,91
4	2,29	2,28	0,46	0,78	2,33	1,60	0,68
5	2,70	2,43	9,97	0,48	2,42	11,80	0,66
6	3,68	3,79	2,99	0,59	3,80	3,30	0,35
RMSRE (%)			21,29			24,15	

Table 27. Optimization results in the output space, obtained with Bayesian optimization. For each mode considered, the updated value and the identified value are reported, as well as the relative error and the MAC value. In the table, the original paper results are reported for reference. Also, the RMSRE is displayed for both the Bayesian results and the original paper results.

Looking at the updated modal properties, the Bayesian optimization approach returns results that are consistent with what obtained in the 2017 study. Generally, natural frequencies obtained through Bayesian optimization show good correlation with the identifies ones. The modes that exhibit the highest amount of error are the third and the fifth, this was already the case in the original study. The MAC values of the first three modes suggest good correlation with the identified mode shapes of the tower, while the last three denote some problems of incoherence (especially the fifth and the sixth). This issue is shared with the earlier analysis, indicating some problems due to the quality of the measurements at the highest modes or the inadequacy of the FE model to capture the dynamic behavior of the bell tower at higher frequencies. Indeed, due to these reasons,

fitting the response due to modal features beyond the third mode is often unpractical in model updating applications (Friswell & Mottershead, 1995).

The results relative to the input space, and so the estimated parameters (that generate the updated modal properties just discussed), are displayed in Table 28. The table reports, for each updating parameter, the estimated value obtained through Bayesian optimization and the estimations of the former damage assessment analysis as reference.

	Bayesian optimization (LCB) 500 fun. eval.	**Original paper** Ref. (2017)
Parameter	**Estimated value**	**Estimated value**
E_1 (GPa)	**0,726**	**0,508**
E_2 (GPa)	**0,792**	**0,806**
E_3 (GPa)	**1,908**	**0,850**
E_4 (GPa)	**3,098**	**4,500**
k_1 (N/m)	5,34E+03	4,13E+06
k_2 (N/m)	4,60E+04	5,05E+02
k_3 (N/m)	1,11E+04	1,60E+06
k_4 (N/m)	5,81E+03	6,34E+05
k_5 (N/m)	6,25E+07	1,50E+09
k_6 (N/m)	8,60E+03	1,06E+05
v (-)	0,46	~

Table 28. Optimization results in the input space, obtained with Bayesian optimization. The estimated value of each updating parameter is shown in the table, along with the estimations of the original paper as reference.

While focusing, at first, on the elasticity moduli of the four sub-structures, that are the parameters of interest as these enable assessing the level of damage of the bell tower, it can be noticed how results coming from the Bayesian optimization approach are mostly in line with the former study, except for the elasticity modulus of the third sub-structure. The low values obtained for the two lowest levels suggest that the bell tower probably endured high levels of damage in these areas, which mostly affect the lower modes of the

building. The only slight difference here is a marginally higher estimate of the second elasticity modulus, which was predicted to show a little more damage in the former study. Instead, a substantial difference can be seen for the elasticity of the walls at the third level of the structure. In fact, the former damage analysis highlighted a much higher level of damage here. Judging by the value estimated through Bayesian optimization, this specific part of the tower was either less damaged by the seismic event, or originally characterized by walls built with stiffer and more qualitative material. This hypothesis is made more plausible by taking into account that the building, which construction started in the late fourteenth century, was severally altered in the seventeenth century, when the height of the bell tower was tripled and the original structure reinforced (Zanotti Fragonara, Boscato, & Ceravolo, 2017). Finally, this time in coherence with the studies from 2017, the walls of the fourth level was found to be significantly stiffer than the rest of the structure. This result, shared by both updating procedures, is not easily explainable, perhaps it may be ascribed to what speculated just before.

Regarding the springs elements which model the degree of constrain enforced by the adjacent structures, the results of the Bayesian optimization approach agree with the estimations of the former analysis, particularly for what concerns the value of k_5, which stands out in both cases with respect to the other link-element parameters, by an order of magnitude of 10^3. The greater stiffness of the fifth spring element suggests that the architectonical element that has the most impact on the dynamic response of the bell tower is the easternmost apse arch, which appears as the most constraining one. All the other elements (particularly the ones modeled by k_2, k_4 and k_6, hence the nave walls and the rectory wall) seem provide little containment at the bell tower.

Besides all the considerations just made, for the reasons afore discussed, there's really no way to establish which of the two analyses returned the best results (which are, nevertheless, largely consistent), and more in general, there is no mean to assess the level of validity of these kinds of damage assessment methods, aside from good agreement with visual inspections. This model updating setup was in fact made very difficult by a non-ideal placement of the measurement instrumentation, which prevents from achieving reliable damage estimations of the fourth level of the bell tower.

5 Conclusions

Finite element model updating is an expensive inverse problem, particularly prone to ill-conditioning. When using iterative methods that draw on penalty functions, one of the most critical aspects is carrying out the optimization task. In fact, these methods most often lead to insidious non-convex penalty functions, which may be extremely difficult to optimize. Moreover, as the need to deal with complex FE models frequently arises, one must cope with expensive penalty functions. Hence, the optimization technique constitutes a critical element of the updating procedure, as it should feature good sampling efficiency, global search attitudes, and adequate accuracy.

Overall, the Bayesian approach to optimization proven itself to be admirably suitable for this challenging task, showing quite impressive results in every case-study analyzed. When dealing with numerical data, which enables to isolate the optimization problem from other major critical aspect of model updating, Bayesian optimization stands on its own when compared to other well-established global optimization techniques (namely generalized pattern search, simulated annealing and Genetic algorithms), featuring far superior sampling efficiency, greater accuracy and better aptitude at detecting the global function minimum, particularly as the number of dimensions (and therefore the complexity) of the optimization problem increases. Fewer objective function evaluations translate in shorter optimization times, greatly relieving the end user, as model updating problems often involve trial-and-error procedures. One major drawback of SA and GA is that these techniques need a high number of function evaluations to perform well. Also, these algorithms tend to provide results denoted by poor accuracy. Generalized pattern search algorithm, on the contrary, may exhibit scarce global search aptitudes, as it was either found converging too quickly or failing at spotting the global function optimum.

Concerning the implementation of the Bayesian optimization technique, logarithmic transformation of the input variables was often found to be (at least marginally) beneficial for the quality of the Gaussian Process regression. Similarly, some kernels showed to be

more suitable given a specific updating problem, leading to a better surrogate model validation. While, in general, improvements over the GP validation were not critical, suggesting that none of the four investigated kernel functions would negatively affect the optimization outcome, a higher difference in terms of validation was appreciated for problems identified by a higher degree of complexity. Finally, regarding the acquisition functions, probability of improvement (PI) showed a highly aggressive behavior, exploiting the prediction by not taking into account its uncertainty as given by the probabilistic model, while expected improvement (EI), its variant (EI+) and lower confidence bound (LCB) demonstrated much better explorative attitudes. Among the four, LCB consistently returned the (slightly) better results, attaining the best balance between exploitation and exploration. Using automatic relevance determination (ARD) kernels, it is possible to gather useful information about the problem sensitivity to each parameter by retrieving the hyperparameters (i.e., length scales) of the GP. This sort of "bult-in" sensitivity analysis is particularly advantageous in structural model updating applications, where information about parameters relevance (often scarcely known in advance) can be considerably useful for understanding the structural behavior. Moreover, ARD kernels automatically discard irrelevant dimensions from the optimization procedure.

Also in the last case study, when dealing with experimental data, Bayesian optimization showed to be up for the task, providing interesting results about the damage assessment of the bell tower, that are mostly in agreement with the original study. Of course, when using real modal data, one must face the other major critical (arguably more than the optimization problem) aspects of model updating, namely FE model limitations and experimental data reliability. In fact, in model updating applications involving real modal data, the global function minimum may not even represent the best physically meaningful solution to the problem. As many local minima solutions may still be equally eligible, combinatorial optimization approaches, that aim at obtaining optimal results from several competing optimization strategies, may be employed. For an example of such an approach, see (Miraglia, Lenticchia, Ceravolo, & Betti, 2019).

While demonstrating impressive performance, there are some aspects of the Bayesian approach implementation used in this work that may be improved or further investigated. Firstly, in this implementation the fitting of the surrogate model, a Gaussian Process, occurs at every algorithm iteration. When fitting a GP, the set of optimal kernel hyperparameters is selected by maximizing the marginal log-likelihood, which obliges to compute several thousands of predictions. This operation presents non-negligible computational complexity when the number of observations used to fit the GP is relatively high. Therefore, optimizing the hyperparameters at the initial seed only, and holding them constant throughout the rest of the optimization process, could significantly improve optimization times. In addition, as at some point of the optimization procedure the acquisition function will start selecting sampling points very close to each other, laying about the alleged optimum point, the Gaussian Process regression may become increasingly "biased" if we keep on optimizing the hyperparameters. Another aspect, which application in model updating problems can be investigated, is the use of an alternative surrogate model. Indeed, despite being the most used probabilistic models, Gaussian Processes show bad scalability as the number of observations grows. Many surrogates have been successively employed by researchers: the implications on the Bayesian optimization performance could be evaluated when it comes to structural model updating problems.

6 References

Ahmadian, H., Gladwell, G. M., & Ismail, F. (1997). Parameter Selection Strategies in Finite Element Model Updating. *Journal of Vibration Acoustics*, 119(1): 37-45.

Alvin, K. F. (1996). Finite Element Model Updating via Bayesian Estimation and Minimisation of Dynamic Residuals. *Proc of the 14th Intl Modal Analysis Conf.*, (p. 428–431).

Arnab, D., & Chakrabarti, B. K. (2005). *Quantum Annealing and Related Optimization Methods.* Berlin : Springer-Verlag Berlin Heidelberg.

Atamturktur, S., & Laman, J. A. (2012). Finite element model correlation and calibration of historic masonry monuments: review. *Struct. Design Tall Spec. Build.*, 21: 96-113.

Bassoli, E., Vincenzi, L., D'Altri, A. M., de Miranda, S., Forghieri, M., & Castellazzi, G. (2018). Ambient vibration-based finite element model updating of an earthquake-damaged masonry tower. *Struct Control Health Monit.*, 25:e2150.

Biswa, N. D. (2002). Finite-Element Model Updating, Eigenstructure Assignment and Eigenvalue Embedding Techniques for Vibrating Systems. *Mechanical Systems and Signal Processing.*, 16(1):83-96.

Bobak Shahriari, K. S. (2016). Taking the Human Out of the Loop: A Review of Bayesian Optimization. *Proceedings of the IEEE, 104*(1).

Boscato, G., Ceravolo, R., Pecorelli, M. L., Ponzalino, S., Russo, S., & Zanotti Fragonara, L. (2013). Sensitivity analysis of damaged monumental structures: the example of S. Maria del Suffragio in L'Aquila. *AIMETA* (p. 1-10). Torino, Italy: Edizioni Libreria Cortina.

Bull, A. D. (2011). Convergence rates of efficient global optimization algorithms. *Journal of Machine Learning Research*, 12:2879-2904.

Ceravolo, R., Pistone, G., Fragonara, L. Z., Massetto, S., & Abbiati, G. (2016). Vibration-based monitoring and diagnosis of cultural heritage: a methodological discussion in three examples. *Int. J. Archit. Herit.*, 10:375–395.

Chen, J. C., & Garba, J. A. (1980). Analytical Model Improvement Using Modal Test Results. *AIAA Journal*, 18:684–690.

Cox, D. D., & John, S. (1992). A statistical method for global optimization. *Proc. IEEE Conference on Systems, Man and Cybernetics*, *2*, p. 1241–1246.

Fox, R. L., & Kapoor, M. P. (1968). Rates of change of eigenvalues and eigenvectors. *AIAA Journal*, 6(12):2426-2429.

Friswell, M. (2008). Damage Identification using Inverse Methods. In A. Morassi, & F. Vestroni, *Dynamic Methods for Damage Detection in Structures.* (p. 13-66). Vienna: Springer.

Friswell, M. I., & Mottershead, J. E. (1995). *Finite Element Model Updating in Structural Dynamics.* Dordrecht: Springer, Dordrecht.

Friswell, M. I., & Penny, J. E. (1992). Assessing Model Quality in Parameter Updating Procedures. *10th International Modal Analysis Conference*, (p. 188-194). San Diego, California, USA.

Friswell, M., & Mottershead, J. E. (2001). Inverse Methods in Structural Health Monitoring. *Key Engineering Materials*, 204-205(204):201-210.

Golano, P. G., & Zanotti Fragonara, L. (2018). Numerical and Experimental Modal Analysis Applied to an Optical Test System Designed for the Form Measurements of Metre-Scale Optics. *Shock and Vibration*, vol. 2018, 14 p.

Hansen, P. C. (1998). *Rank-Deficient and Discrete Ill-Posed Problems: Numerical Aspects of Linear Inversion.* Philadelphia, PA, US: SIAM.

Holland, J. (1992). *Adaptation in Natural and Artificial Systems.* Ann Arbor, US: University of Michigan Press.

Hooke, R., & Jeeves, T. A. (1961). "Direct Search" Solution of Numerical and Statistical Problems. *Journal of the ACM*, 8(2):212-219.

Hutter, F., Hoos, H. H., & Leyton-Brown, K. (2011). Sequential model-based optimization for general algorithm configuration. In *Learning and Intelligent Optimization.* (p. 507–523). Springer.

Ingber, L. (2000). Adaptive simulated annealing (ASA): Lessons learned. *Control and Cybernetics*, 25:33-54.

Jones, D. R. (2001). A taxonomy of global optimization methods based on response surfaces. *J. Global Optimization*, 21:345–383.

Jones, D. R., Schonlau, M., & Welch, W. J. (1998). Efficient Global Optimization of Expensive Black-Box Functions. *Journal of Global Optimization*, 13:455–492.

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by Simulated Annealing. *Science*, 220(4598):671-80.

Kushner, H. J. (1964). A new method of locating the maximum of an arbitrary multipeak curve in the presence of noise. *J. Basic Engineering*, 86:97–106.

Kushner, H. J. (1964). A new method of locating the maximum of an arbitrary multipeak curve in the presence of noise. *Journal of Basic Engineering*, 86:97–106.

Lallement, G., & Piranda, J. (1990). Localization methods for parametric updating of finite element models in elastodynamics. *Proceedings of the 8th International Modal Analysis Conference*, (p. 579–585).

Levin, R. I., & Lieven, N. A. (1997). Dynamic finite element model upsating using simulated annealing and genetic algorithms. *Mechanical Systems and Signal Processing*, 12(1):91–120.

Maiworm, M., Limon, D., & Findeisen, R. (2021). Online learning-based Model Predictive Control with Gaussian Process Models and Stability Guarantees. *International Journal of Robust and Nonlinear Control*.

Marwala, T. (2010). *Finite Element Model Updating Using Computational Intelligence Techniques.* London: Springer-Verlag London.

Matérn, B. (1960). *Spatial Variation.* (1986 ed.). Springer-Verlag New York.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., & Teller, E. (1953). Equation of State Calculations by Fast Computing Machines. *Journal of Chemical Physics*, 21(6):1087-1092.

Miraglia, G., Lenticchia, E., Ceravolo, R., & Betti, R. (2019). Synergistic and combinatorial optimization of finite element models for monitored buildings. *Struct. Control Health Monit.*, 26:e2403.

Mockus, J. (1982). The Bayesian Approach to Global Optimization. In R. Drenick, & F. Kozin, *System Modeling and Optimization* (p. 38:473–481). Berlin: Springer Berlin - Heidelberg.

Mockus, J. (1989). *Bayesian Approach to Global Optimization.* Amsterdam: Springer Netherlands.

Mockus, J. (1994). Application of Bayesian approach to numerical methods of global and stochastic optimization. *J. Glob. Optim. 4*, 347–365.

Mockus, J., Tiesis, V., & Zilinskas, A. (1978). The Application of Bayesian Methods for Seeking the Extremum. In L. C. Dixon, & G. P. Szegö, *Towards Global Optimization* (Vol. 2, p. 117–129). Amsterdam: Elsevier.

Mottershead, J., & Friswell, M. (1993). Model Updating In Structural Dynamics: A Survey. *Journal of Sound and Vibration*, 167:347-375.

Neumaier, A. (1998). Solving Ill-Conditioned and Singular Linear Systems: A Tutorial on Regularization. *SIAM*, 40(3):636–666.

Rasmussen, C. (2004). Gaussian Processes in Machine Learning. In O. Bousquet, U. v. Luxburg, & G. Rätsch, *Advanced Lectures on Machine Learning.* (p. 63-71). Berlin: Springer Berlin.

Rasmussen, C. E., & Williams, C. K. (2006). *Gaussian Processes for Machine Learning.* Cambridge, Massachusetts: MIT Press.

Sacks, J., Welch, W. J., Welch, T. J., & Wynn, H. P. (1989). Design and analysis of computer experiments. *Statistical Science*, 4(4):409–423.

Sasena, M. J. (2002). Flexibility and Efficiency Enhancement for Constrained Global Design Optimization with Kriging Approximations. *PhD thesis, University of Michigan*.

Sehgal, S., & Kumar, H. (2016). Structural Dynamic Model Updating Techniques: A State of the Art Review. *Arch Computat Methods Eng*, 23, 515–533.

Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., & de Freitas, N. (2016). Taking the Human Out of the Loop: A Review of Bayesian Optimization. *Proceedings of the IEEE*, 104(1):148-175.

Shahverdi, H., Mares, C., Wang, W., & Mottershead, J. E. (2009). Clustering of parameter sensitivities: examples from a helicopter airframe model updating exercise. *Shock and Vibration*, 16 (1): 75–87.

Simoen, E., De Roeck, G., & Lombaert, G. (2015). Dealing with uncertainty in model updating for damage assessment: A review. *Mechanical Systems and Signal Processing*, 56:123-149.

Snoek, J., Rippel, O., Swersky, K., Kiros, R., Satish, N., Sundaram, N., . . . Adams, R. P. (2015). Scalable Bayesian optimization using deep neural networks.

Proceedings of the 32nd International Conference on Machine Learning, (p. 2171–2180).

Springenberg, J. T., Klein, A., Falkner, S., & Hutter, F. (2016). Bayesian optimization with robust bayesian neural networks. *Advances in Neural Information Processing Systems*, (p. 4134–4142).

Streltsov, S., & Vakili, P. (1999). A non-myopic utility function for statistical global optimization algorithms. *J. Global Optimization*, 14:283–298.

Wang, D., Tan, Z., Li, Y., & Liu, Y. (2013). Review of the Application of Finite Element Model Updating to Civil Structures. *Key Engineering Materials*, 574:107-115.

Wang, Z., Gehring, C., Kohli, P., & Jegelka, S. (2018). Batched large-scale Bayesian optimization in high-dimensional spaces. *International Conference on Artificial Intelligence and Statistics (AISTATS)*, (p. 745–754).

Zanotti Fragonara, L., Boscato, G., & Ceravolo, R. (2017). Dynamic investigation on the Mirandola bell tower in post-earthquake scenarios. *Bull Earthquake Eng.*, 15:313–337.

Zimmerman, D. C., & M., K. (1992). Eigenstructure Assignment Approach for Structural Damage Detection. *AIAA Journal*, 30:1848–1855.