POLITECNICO DI TORINO

Corso di Laurea Magistrale in Ingegneria Aerospaziale

Tesi di Laurea Magistrale

Multifidelity active learning for aerospace design and optimization

Relatore:
Dott. Ing. Laura Mainini

Correlatore:
Prof. Paolo Maggiore

Candidata:
Michela Nardelli

Anno accademico 2021/2022

A mamma, papà e Raffaella,
il mio cielo in una stanza.

Abstract

Real-world engineering optimization problems are often extremely complex and onerous, in terms of time and computational costs. In aerospace engineering, these problems may concern fluid dynamics analysis, structural and system design and integration, or their combination in a multidisciplinary context: multiple information sources need to be optimally handled. Analysing the single discipline already results expensive, therefore, extending the analysis into a multidisciplinary framework becomes frequently unaffordable. However, a trade-off among different configurations of design is useful to guide the decision making in the preliminary stages of a project. For these reasons, advanced methods to accelerate the assessment of design alternatives are demanded. Possible approaches are based on the use of faster models, also called *surrogates*, that are employed to provide approximations of physical responses to guide the design and optimization process. Within the context of using faster and cheaper models, multifidelity methods act leveraging the representation of the physics at different levels of abstraction. The main objective is the use of less expensive models for a massive exploration of the design space and the reduction of the high-fidelity evaluations to the minimum amount required to maintain an adequate accuracy. Among the different methods that can be found in literature, this thesis focuses on those which are also referred to as *multifidelity surrogate based optimization*. They aim is to combine information from different level of fidelity into a unique surrogate model which is used to guide the design optimization.
The particular goal of this thesis is to compare and assess different methods belonging to this specific class. The focus will be on methods based on adaptive sampling techniques to improve the surrogate while searching for the optimum. We refer to this as an *active learning scheme* and it has been implemented through a Bayesian framework. Therefore, we will consider three implementations of multifidelity Bayesian Optimization (MFBO) which differ for the acquisition functions, that are: Multi-Fidelity Expected Improvement (MFEI), Multi-Fidelity Probability of Improvement (MFPI) and Multi-fidelity Max-value Entropy Search (MFMES). We demonstrate the performance of the three methods over benchmarks problems of class L1 and L2 determined by an international community of experts and partitioners of multifidelity methods. They are set as standard test cases to assess different multifidelity formulations and methodologies. We consider six L1 problems, which are analytical functions cheap to evaluate and whose optimum is known; the L2 problem has relevance for aerospace applications and relates to the optimization of an airfoil shape to minimize the drag coefficient in transonic conditions. Our specific implementation of those methods leads to find the global minimum of the objective function, by using just two different fidelity levels. For the L1 applications, the low- and the high-fidelity models are formulated following the guidelines of the international research group while, for the L2, the models differ for the coarseness

of the grid for the aerodynamic analysis. The refinement of the experiment setups have been done by combining both the experts' opinions and a trial and error approach. The results obtained demonstrate that the MFEI and MFPI have the best performance, even with a narrow initial dataset, while the MFMES generally needs a greater set.

Summary

List of Figures

List of Tables

Chapter 1

Introduction

In aerospace industry, collecting data for design optimization can be very expensive. This because, information from several fields need to be joined. Designing an aircraft or a spacecraft, leads to the necessity to efficiently combine structural, subsystems, aerodynamic, flight test data, just to name a few, to achieve the best configuration. At the same time, comparing more than one configuration may result interesting to select the optimal solution. It is really important for the early phases of design, because it can guide the decision making towards a reduction of time and resources required for the design cycle. However, handling this big amount of data and their interconnections, may be impossible. Machine learning can help addressing these problems for the possibility to learn a cheaper surrogate model. These techniques have become popular after the greater computational performances achieved by modern calculators and the investments made in this new field. Generally, they pass through the realization of emulators capable to model accurately the problem, combining different sources of information in a unique source, on the basis of the available data, with the great advantage of saving resources. Building an *emulator*, also called *surrogate model, metamodel, response surface model*, is useful to replace traditional mathematical model in contexts like that mentioned before, in which either a model may not be available or its analytical solution does not exist. A surrogate is a simplified approximation model, built by observing the first data collected at the initial set of points, randomly picked via Design Of Experiment (DOE). In this way, it results capable to predict the outputs of the system of interest at points not yet sampled, reducing the number of evaluations of the true and expensive function. Once the surrogate is ready, the next step concerns solving the optimization problem. The Surrogate Based Optimization (SBO) derives from this framework, indeed, it adopts surrogates for quickly finding the global or local optimum of a problem. The performance obtained from this process, depends on the accuracy of the surrogate, which is improved by evaluating the true function in points efficiently selected with a sampling strategy.
Sampling methods for data-driven optimization can be static or adaptive. The first uses offline/online techniques, the latter is based on active learning (also called adaptive learning). Offline/online techniques use a dataset, stored before the optimization process starts, from offline information sources upon which they build the surrogate online: the optimization is guided only by the surrogate, because a new collection of data, to update it, is not generated during the current optimization. Therefore achieving good performance lies on the reliability with which the surro-

gate has been implemented [1]. The emulator can be, then, improved with data stored offline from previous optimization processes. It may require a large amount of training points to achieve the best solution and it may lead to onerous and unaffordable computational costs [2]. Whereas, for what concerns active learning, it builds an initial surrogate following the same algorithm of offline/online techniques, but for the next iterations, it uses only the most informative data about the difference between model estimates and real objective function observations. These data are progressively obtained during the sampling, to leverage the surrogate and interactively guide the search path towards the optimal fitting. Hence, the surrogate model learns from observations, as they are collected. To summarize, adaptive sampling implies predicting which are the most informative data and where they are located in the design domain, on the basis of the objective observations assimilated during the computation. The Bayesian Optimization (BO) implements the active learning techniques via SBO. A Bayesian framework is suitable to evaluate black-box functions: the objective functions have not a known mathematical formulation, therefore, only their inputs and their outputs can be handled. Hence, BO assumes the objective function as a random function and uses a *prior* distribution, combined with the first data available, to predict a *posterior* distribution. Then, it is adopted to build an acquisition function, that guides the algorithm research towards the global optimum. A optimization manages adaptive response surfaces in order to reduce the number of observations required. Superior performances can be achieved modeling a surrogate with a combination of model with different fidelity levels, i.e. with different degree of accuracy. This allows to reduce further the computational efforts required for building the surrogate, mitigating the loss of accuracy introduced with the use of a surrogate model in place of the costly high-fidelity representation. In this case, we talk of Multifidelity Bayesian Optimization (MFBO).
This thesis focuses on this type of adaptive learning and aims to implement and characterize three different multifidelity Bayesian methods that differ for the acquisition functions implemented. Multifidelity Expected Improvement (MFEI), Multifidelity Probability of Improvement (MFPI) and Multifidelity Max-value Entropy Search (MFMES), whereas Multifidelity Predictive Entropy Search (MFPES) is just analysed, but has not been implemented, due to incompatibility with the Gaussian Process model adopted. They are conceived with the aim to guide the selection of the next point to better infill the design space, by maximizing the expectation or the probability of improvement, i.e. the gain of information about the objective function, or the maximum entropy variation, considering the current best prediction assessed by the model.
Multifidelity methods, have been applied to different aerospace problems in the last decades. They have been tested for optimization of airfoils aerodynamics, e.g. to maximize the airfoil lift in transonic condition [3, 4]. Passing to wing design leads to a multidisciplinary and interdisciplinary context, in which material science, structural, aerodynamic data for loads production, aeroelastic analysis (e.g. to control flutter problems), or flight controls may affect each others, and constraints on ease of manufacturing, weights or fuel to board, may require to be satisfied simultaneously. About that, it is possible to name the work made by Mainini and Maggiore [5], that follows a multilevel design architecture, capable to recognize the variables that mostly affect the mission requirements, within each field of interest, namely aerodynamics, structures, materials (metals and composites) and manufacturing costs, in

order to use them for solving the optimization problem.
Implementations of multifidelity methods can be found for predicting the drag that affects subsonic aircrafts with also a special attention dedicated to the use of composite structures, active control and new aerodynamic performance in order to mitigate environmental impacts (noise, global emissions, local air quality) [6].
Again within the context of aerospace engineering, several applications concern High Speed Civil Transports (HSCT) [7], from which derive the use of multifidelity surrogate approaches for the optimization of wing structural weights of an HSCT [8, 9]. They also find out a way to reduce the loss of performance (i.e. accuracy and costs) of the surrogates with increasing dimensionality (over 10-15D), identifying reasonable design space within which to limit the response surface domain, so that to preclude unreasonable designs. Then, other applications [10] adopted this strategy for minimizing the take off gross weight of HSCT. Furthermore, to remain on the high speed overview, it is possible to name also the studies [11, 12, 13], which aim to optimally manage aerodynamic performance and the loudness of the sonic boom at the ground. Also shaping of blended wing body Unmanned Aerial Vehicle (UAV), using adaptive multifidelity methods, can be found in literature [14] with the aim of reducing drag coefficient. Tappeta, Nagendra and Renaud [15] use surrogates for emulating the different disciplines, with which they face, for the optimization of aircraft engine components (e.g. turbines,compressors) exposed to high temperatures. Other types of implementations can be found for rotorcrafts [16, 17], because of the strong interdisciplinary of this sector, due not only to coupled aerodynamics, dynamics and control systems, but also to the necessity to reduce vibrations.
For what concerns space design applications, multifidelity methods are adopted for designing re-entry capsules [18], therefore in an unmanned mission background [19], considering also the problem of building accurately a thermal shield and distributing the mass to not affect aerodynamic performances [20], or adding the assessment of optimal trajectories that could reduce the propellant mass required [21]. In literature, implementations on launch vehicles can be found [22], that also have to deal with stages layout and mass partitioning.
However, to briefly expand the discussion beyond design problems, it is possible to observe several other applications in aerospace, which may include remotely piloted drones, diagnostic and prognostic, e.g., to name a recent work, controlling a spacecraft attitude stabilization in presence of actuator fault, parameter uncertainty and external disturbances [23].
In this thesis the work is shaped around the aim of testing the methods selected on benchmark problems, in order to characterize their mathematical features. The purpose is to experimentally collect data about the methods performances. This assessment is necessary, because the multifidelity formulation of these methods is recent and has not yet been widely explored. The methods are applied to standard problems proposed by a specialised international group. We will employ those which are also referred to as L1 and L2 problems, that have aerospace relevance. The first are simplified analytical problems with known mathematical behaviours, the last are more articulate and expensive problems closer to real-life engineering applications. These test cases are selected for stressing the methods on multimodal problems and with increasing dimensionality, so as to evaluate their strengths or weaknesses. Then, the methods are tested on a L2 aerodynamic problem concerning the shape optimization of an airfoil in transonic regime, with the aim of minimizing the drag

coefficient. The profile is modeled parametrically through an open source software, which includes either the solver to use for Computational Fluid Dynamics (CFD), SU2, or the mesh generator GMSH, which is called to run at every new airfoil design found by the optimization algorithm.
Chapter 2 will show the landscape in which surrogate models, in single and multiple fidelity are developed, considering interpolation and regression methods to construct them, and showing some sampling strategies that adopt DOE tools to efficiently choose the starting dataset. Chapter 3 extends the use of the response surfaces into a Bayesian framework, describing the advantages of this method and focusing on the analysis of the single-fidelity acquisition functions. In Chapter 4, these functions are extended to a multifidelity context. Therefore, it is showed how the algorithms have been implemented. Chapter 5 includes a comparison among the methods performance, thanks to the application on L1 problems that include: Forrester function [24], Sinusoidal squared function, Clark and Bae 1D, 2D and 3D functions [25], Rosenbrock 2D and 5D functions, Shifted-Rotated Rastrigin function and spring-mass system 4D. Chapter 6 explores the methods performance on the L2 aerodynamic problem, describing the physics of the problem, its formulation and the setup (not trivial to select) for the experiments.

Chapter 2

Surrogate Based Optimization

Aerospace engineering often has to deal, even in the preliminary design stages, with complex systems that neither can be easily computed with physical models, nor with empirical tools. It is the case of aerodynamic shape optimization, structural design and multidisciplinary optimization (MDO) [26, 27], which can simultaneously connect more disciplines, such as propulsion, flight controls, aerodynamics, aeroacoustics and structures in a perspective of design of aircrafts or spacecrafts. Furthermore, making a superior technological design can be a strategic advantage in aerospace industries, whose development is also based on competitiveness. Especially in the preliminary phases of a project, and for guiding the decision making process, it is essential the evaluation of numerous and innovative configurations to extensively understand the complex phenomenology of a physical problem. All these scenarios lead to manage a large amount of data and unknown variables interconnections. Experimental data are costly to obtain and complex behaviours are difficult to characterize only by using an experimental approach. For these reasons, simulations are used to translate, in digital world, the interdependence among physical factors. Then, the optimizer aims to use these models to inform the design process. It is not always possible to adopt models which can describe the reality with the highest level of accuracy, even if they can provide a more reliable design. This is due, not only to the lack of an analytical formulation of the problem, but also to the non negligible computational costs of the high-fidelity models. They, indeed, may encompass Computational Fluid Dynamics (CFD), finite-element structural models or physical experiments, e.g. in wind tunnels. At the same time, it is not feasible to simply replace the high-fidelity model with a low-fidelity one (i.e. with less accuracy than the corresponding high-fidelity model), because it could lead to unsatisfactory approximations and designs. Hence, different approaches exist to accelerate the computation of the same output assessed by the high-fidelity function, via lower-fidelity models also called *surrogates, meta-models, response surfaces, emulators*. They can help in alleviating numerical noise or discontinuities in the objective function.
Different types of surrogate exist, in particular, according to Eldred et al. [28], they are classified as: *data-fit, hierarchical* and *reduced-order* models.
The data-fit models are frequently computed to be computationally efficient, smooth and as much as possible accurate, using the high-fidelity model as a guide to iteratively improve themselves. Data-fit surrogates are especially preferred, when the high-fidelity function is a black box, whose structure is not known. Indeed, these models can handle inputs and outputs of a black box function, that are its only

available information. They describe analytically the data collected from the original model, without providing a physical meaning. The coefficients of the linear combination of data-fit models are evaluated via interpolation or regression techniques. This choice affects the accuracy of the fitting. Polynomials or radial basis functions are widely used methods. The latter can find efficiently these coefficients, if they are based on the Gaussian density function [29]. Kriging (a stochastic process approach to approximating functions) is a particular application of data-fit models. More details about how to fit a model to data, can be found in Section 2.2.
Hierarchical models, also called *multifidelity or variable fidelity* models, employ the strategy of simplifying high-fidelity models. The simplification can concern, for example: neglecting nonlinear terms, reducing dimensionality, simplifying physics models (e.g. the boundary conditions) or geometry, coarser grid discretization of domains or implementing early stopping criteria, so as to settle for just partially converged results [30]. They are still physics-based models [28].
Reduced-order models (ROM) [31], or *projection-based models* [29], aim to identify a low-dimensional subspace which is representative of the main features of the system of interest. It is, generally, an intrusive method, because employs a model reduction to project the original high-dimensional system into lower-dimensional subspace.
In this framework, surrogate models can be useful to solve optimization problems, because they can achieve reasonable approximations of the high-fidelity model, at least locally, thus substantially reducing time and computational costs. Generally, an optimization problem can be expressed in the form:

$$\min_{\boldsymbol{x}\in\mathcal{A}} f(\boldsymbol{x})$$

where $\mathbf{x} \in \mathbb{R}^d$ denotes the input vector of dimension d, $\mathcal{A}$ is a set of possible inputs, and $f(\boldsymbol{x}) \in \mathbb{R}$ is the continuous objective function. Traditional optimization methods evaluate the objective function at given points till a stopping criterion is met, but in design contexts like that mentioned before, this computation can be really expensive. These methods can become inefficient when a considerable number of evaluations is required, so a *Surrogate Based Optimization* (SBO) approach is preferred. In optimization contexts, the objective function may be a black-box, that is a nonlinear and non-convex function f, whose derivatives are unavailable and whose closed-form expression is unknown. Hence, basing the optimization process on the use of a surrogate model, allows to synthesize sufficient information about the black-box function and lowers the amount of required function evaluations thus saving time, computational resources and costs. Surrogates can be built following an iterative optimization process, *active learning* (or *adaptive learning*) technique which, at each iteration, chooses the best informative points in which the high-fidelity model needs to be evaluated to update the surrogate. Therefore, adaptive strategies dynamically learn how to efficiently seed points into the design space, progressively training the model. This learning strategy goes against the online/offline techniques, that use a set of data collected previously offline, to build online the surrogate.
To summarize, the main steps of a surrogate-based optimization framework include:

1. Choose the variables to optimize, i.e. the design variables. It may not be trivial for new design problems, therefore the choice may be guided by the expert experiences or by the information obtained from the first experiments, that can be used to iteratively improve the selection. However, the feasible amount of variables is mainly influenced by the available resources.

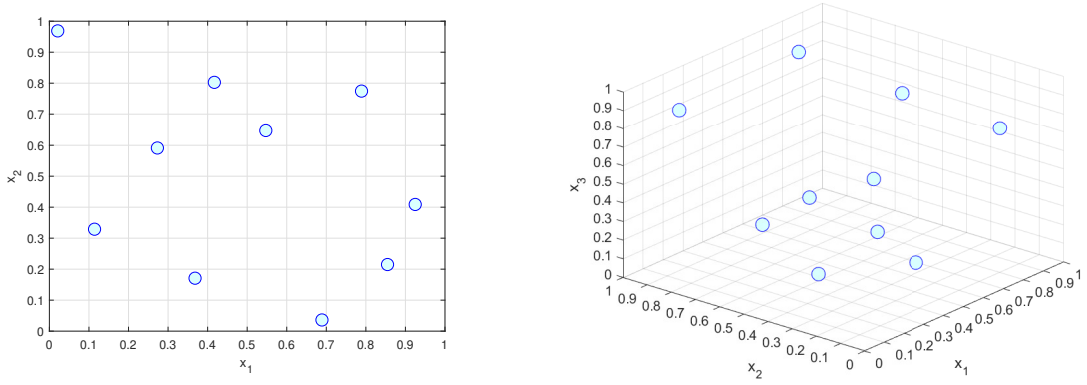
2. After the identification of the space of design, the interval within which the design variables change needs to be set. The Design Of Experiment approach is used to select a suitable sampling strategy for the initial points to query with the surrogate.

3. A surrogate is built on the basis of the first observations.

4. A search for the optimum is run, by using the surrogate, and gives in output new design points (*infill points*) at which to assess the objective function. In this way, information are collected and are iteratively provided to the surrogate, in order to validate and train it towards the best solution (adaptive sampling) [32].

2.1 Design Of Experiment (DOE)

The selection of initial points (*training dataset*), at which the models will be evaluated, is computed using Design Of Experiment (DOE) tool. As a rule of thumb, Morris and Mitchell proposed to approximately choose the number n of initial samples points, for the low- and high-fidelity models, according to: $n = 10k$, where k represents the dimensionality (i.e. the number of variables) of the problem [33]. However, the particular choice of strategy to use for allocating samples in the design space, lies on the number of samples affordable with the resources available and on the technique selected to construct the model [34].
Design Of Experiment refers to the process of planning, implementing appropriate choice of design and statistically analysing an experiment, in order to achieve valid and objective conclusions, reducing the number of samples required. It is an essential technique for quickly improving process performances [35], because it aims to maximize the amount of initial information, thus creating the training data set upon which the surrogate will be constructed [36].
Design of experiments is used to decide which observations provide more useful information to build a model; on the other hand, it is used to investigate the sensitivity of output values with respect to the combination of inputs. This is also known as *sensitivity analysis*. DOE allows to determine the data to use and improves the quality of the information, while eliminating redundant data. This leads to improve the construction of a model in a cost-effective and less time-consuming framework.
These statistical techniques start with a screening phase, which involves the identification of the most important input factors that may affect the system performance and, then, end with an optimization phase (that may use Response Surface Methodology, RSM) [37]. Among the different design methods adopted for screening, it is possible to include, just to name a few: the *Full-Factorial Designs* [38], which measure response variables using all possible combination of factor levels, so they collect data at the vertices of a cube in k dimensions (k is the number of factors adopted); *Fractional-Factorial Designs* which collect data from a specific subset of all possible combinations, determining which factors or their combination, influence more the output. Hence, it is used when there is a large amount of design variables. Also *Central Composite Design* [39] is used, and is an extension of the design points that can be selected with a fractional factorial design: new sampling points are added on the faces of the hypercube and at its center.



(a) 2D LHS

(b) 3D LHS

Figure 2.1: Latin hypercube design

Typically, when no prior knowledge is available about the objective function, e.g. before building a surrogate, space-fillings designs are recommended, because they can spread out the design points nearly evenly throughout the design domain, in order to extract trend information and to reduce bias error (errors that arise when a substantial difference between the trend estimated and the real one, is evaluated). Therefore, they are able to capture the different behaviours of responses in different regions of the experimental space, avoiding replicate runs and giving information about the unknown form of the model.
A famous space-filling design is the Latin hypercube [40], proposed in 1979 and is, even now, one of the most commonly used methods. It consists in a n x k matrix in which, each column is a random permutation of the levels $1, 2, ..., n$, where n represents the number of runs, i.e. the number of points to spread, for k factors $\boldsymbol{x}$. Hence, a level, or bin, indicates the n equally-spaced and non-overlapping intervals by which, each factor is divided within its range [41]: so there are n^k bins. Therefore, a row contains specific values of each of the k-th variables to be used on the n-th model run [42]. One value from each interval is randomly selected with respect to the probability density in the interval, then the algorithm is repeated till all the n values for each input variable k are selected. The n values thus obtained for $\boldsymbol{x_1}$, are then randomly paired to that selected for the other variable $\boldsymbol{x_2}$. These n pairs are combined with the n values sampled from $\boldsymbol{x_3}$, to form n triplets, and so on, until n k-tuplets are formed. Essentially, it is a generalization, to a multidimensional space, of a Latin square: each factor permutation is arranged in such a way that no orthogonal (row or column) contains the same number twice: for all one-dimensional projection of samples and bins, there is only one sample in each bin [34].
There are many possible choices for Latin hypercube design, but results interesting describe the method proposed by Morris and Mitchell [33], in 1995. It seems to be a widely used technique for computer experiments, because of its simplicity and availability in software packages. The algorithm adopted to seed design points, pursues the objective of maximizing the minimum distance between them, assuring a quite evenly sampling. The criterion conceived by Morris and Mitchell, is the following:

$$\min_{\boldsymbol{D}} \left\{ \sum_{i=1}^{n=1} \sum_{j=i+1}^{n} \frac{1}{d^k(x_i, x_j)} \right\}^{1/k}$$

where $\boldsymbol{D} = \{\boldsymbol{x}_1 ..., \boldsymbol{x}_n\}$ is the experimental design area and k is suggested to be chosen in order to be the smallest value that gives a maximin distance design. Figure 2.1 shows an example of Latin hypercube design (*LHS* stands for Latin Hypercube Sample) with ten runs and two and three factors. As explained before, it is possible to notice how, each row and each column are filled with only one sample.
Randomisation is a powerful method for reducing the effect of experimental bias [35]. Sometimes there could be particular situations where randomisation is difficult to be performed due to cost and time constraints, so it might be more appropriate to change that particular factor level less frequently than others. The *restricted randomisation* can be adopted under these circumstances. Therefore, the Latin hypercube is a *quasi-random* method, which differs from a *random sampling* method. It leads to point sets with a discrepancy (i.e. how much the distribution of samples differs from an ideal uniform distribution [43]) smaller than that expected for a pure random sampling, where all subsets of a sampling frame have an equal probability of being selected. Also quasi-Monte Carlo sampling ascribes to this category, and is also a space-filling design method. It uses deterministic algorithm to generate samples into the design domain, in order to achieve a quite uniform sampling. It differs from Pseudo-Monte Carlo, or just Monte Carlo, sampling, because this one implements a pseudo-random points distribution.
Another sampling technique can be the *D-optimal design* where the design point are selected by statistical methods. It seeks to minimize the covariance of the parameters for the current model analysed. Generally, it is used in substitution of traditional methods, like that mentioned before (Full or Fractional Factorial Designs, Response Surface Designs). They are appropriate to calibrate linear models in experimental problems which require sufficiently unconstrained factor in the design space. However, when the model is necessarily nonlinear or certain combination of factors results particularly expensive or complex to be measured, the D-optimal design is a powerful technique to choose, to override these problems.

2.2 Fitting a Response Surface

After selecting the DOE technique to sample the training set, it is necessary to choose a fitting strategy to model the surrogate (Response Surface Methodology, RSM). It generally lies on two basic assumptions [32]:

1. The surrogate models a continuous function. This is a founded assumption, even if in aerodynamic contexts it may fall. For example, in region of shocks, it is well known the presence of sudden discontinuities. This can be avoided by using a combination of multiple surrogates (e.g.[44]) or adding different assumptions.

2. The objective function needs to be smooth.

Further assumptions may concern the shape of the objective function, but especially in presence of black-box functions, it is not trivial.
Generally two branches can be found in building a surrogate model: regression and interpolation techniques. The first includes polynomial regression and neural networks, the latter involves the use of splines (or radial basis functions) and Kriging-based models.

2.2.1 Regression based models

For polynomial regression models [45], the first step requires to find a suitable approximation for the relationship between response (i.e. the output of the system of interest) and independent variables (the factors): a polynomial equation is used. The simplest way to adjust a response surface to data is adopting a linear regression. Assume that $y^i = y(\boldsymbol{x}^{(i)})$ represents the function value computed at the sampled factors $\boldsymbol{x}^{(i)}$ for $i = 1, \ldots, n$, where n is the number of points selected and K the amount of polynomial basis functions f_h. The observations of the objective function, can be expressed as:

$$y(\boldsymbol{x}^{(i)}) = \sum_{h}^{K} \beta_h f_h(\boldsymbol{x}^{(i)}) + \epsilon^{(i)} \tag{2.1}$$

where the β_h's are the unknown regression coefficients to be estimated, and the $\epsilon^{(i)}$'s are error terms, observed in the response, which are supposed to be normally distributed, with mean zero and variance σ^2. The function $f_h(\boldsymbol{x}^{(i)})$ can be a linear function of the independent variables, so in this case, the function that approximates the response, is a first-order model; whereas it can be a nonlinear function, such as a second-order model or higher degree, and the response will be modeled with a curved surface. Almost all RSM problems use first-order, second-order model or both, but is is important to emphasize that it is unlikely that a polynomial model will be a perfect-suited approximation of the true functional relationship among the factors. It can work quite well on a relatively small region of design [36]. Therefore, polynomial surrogates are not employed for high-dimensional, highly multi-modal and non linear problems, also because it is not affordable to evaluate the great amount of data necessary to estimate the terms of the polynomials [32].
Furthermore, choosing the order K of the polynomial approximation is not trivial, because even if selecting greater values of K (i.e. expanding further the Taylor series) yields to more accuracy, at the same time the problem of over fitting the data, including any noise, arises. Several techniques lead to adequately choose the order of approximation K, however further information can be found in [46]
The parameters of the approximating polynomials, can be estimated by the method of least squares. By introducing matrix form to Equation 2.1, what is obtained is $\boldsymbol{f} = \boldsymbol{X}\boldsymbol{\beta}$, with $\boldsymbol{f} = [f(\boldsymbol{x}^{(1)})\ f(\boldsymbol{x}^{(2)})\ \ldots\ f(\boldsymbol{x}^{(K)})]^T$, $\boldsymbol{\beta} = [\beta_1\ \beta_2\ \ldots\ \beta_K]^T$. According to least squares approximation, $\boldsymbol{\beta}$ can be found as :

$$\boldsymbol{\beta} = (\boldsymbol{X^T X})^{-1} \boldsymbol{X^T f} \tag{2.2}$$

which represents the maximum likelihood estimate of β.
However, one of the most used technique to model the response of a deterministic experiment is a Gaussian Process (GP). Deterministic models may require high order terms so, the number of parameters to estimate can be large. A Gaussian Process allows to reduce the number of parameters to evaluate, because it becomes equal to the number of factors. Statistical surrogate models are frequently implemented in the form of a GP, that tries to find a response surface with stochastic processes. Fitting a stochastic process to data, essentially means making a model that summarizes how the function behaves and its properties (e.g. the smoothness) in the design domain and then predicting, with uncertainty, its behaviour at unsampled points, on the basis of the first estimation [47]. These statistical models are preferred precisely for their capability to model complex functions with also an

assessment of their own uncertainty.
Generally, in complex problems, data-driven surrogates are lacking of generalization for the data that are distant from the training dataset. Therefore, Artificial Neural Networks adopt regression models associated to deep learning, thus providing a technique for reducing the amount of objective observations required to make the model more general. The term refers to a system composed of several simple process units, the so called neurons, that work in parallel to perform a particular task. The computational parallelism makes the network really efficient. Hence, a neuron is the basic structure of the network, in which the n inputs $\boldsymbol{x}$ are computed, via non linear operations, in outputs y. A single-layer formulation is showed:

$$y = \frac{1}{1 + \exp(-\eta/T)}$$

where $\eta = w_1 x_1 + ... + w_n x_n + \gamma$ with $w_1 x_1 + ... + w_n x_n$ as the weighted inputs, γ as the bias value of the neuron, and T as a parameter, selected by the user, that determines the shape of the function in terms of more or less steep slope [34]. Among the different ways in which neurons can be combined, one of the most commonly used architecture is that of multi-layer feed-forward network [48]. Modeling a surrogate by means of neural network, requires a first step in which there is the selection of the appropriate architecture, to approximate a set of functions, and a second step in which there is the network training. The last can be handled as a nonlinear least-squares regression problem for a given set of training points [34].

2.2.2 Interpolation based models

In a deterministic problem, the assumption of independent errors, seen before for the least squares estimations, is false because, any lack of fit of the regression model with respect to the data, is due only to modeling errors (incomplete set of regression terms), not measurement error or noise [47]. So, it is more reasonable to assume that error terms are correlated with a weighted (not Euclidean) distance function (Equation 2.3) between the corresponding points $\boldsymbol{x}^{(i)}$ and $\boldsymbol{x}^{(j)}$ as:

$$d(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)}) = \sum_{h=1}^{k} \theta_h |x_h^{(i)} - x_h^{(j)}|^{p_h} \qquad (\theta_h \geq 0, p_h \in [1, 2]) \tag{2.3}$$

where θ_h measures the 'activity' of the variable x_h. It aims to show if a variable is more or less active with respect to the correlation behaviour, that will drops off more or less rapidly. The other parameter, p_h, represents the smoothness of the function along the coordinate direction h. A $p_h = 2$ corresponds to a function with a higher smoothness than a function with value set to one.
The correlation between the errors is:

$$Corr[\epsilon(\boldsymbol{x}^{(i)}), \epsilon(\boldsymbol{x}^{(j)})] = \exp[-d(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)})].$$

Therefore, when the distance between $\boldsymbol{x}^{(i)}$and $\boldsymbol{x}^{(j)}$ is small, the correlation tend to one; whereas it will approach to zero. Thanks to this approximation of the correlation, it is possible to dispense with the regression terms. They can be replaced with a constant term, so what is obtained is:

$$y(\boldsymbol{x}^{(i)}) = \mu + \epsilon(\boldsymbol{x}^{(i)}) \qquad (i = 1, ..., n)$$

where μ is the mean of the stochastic process and $\epsilon(\boldsymbol{x}^{(i)})$ is $\mathcal{N}(0, \sigma^2)$. The term stochastic derives from the error term $\epsilon(\boldsymbol{x}^{(i)})$ that, indeed, is a stochastic process [47]. This is the basis upon which Kriging is built, therefore, it is a combination of a linear regression and a stochastic model fitted to the residual errors, that are modeled following a Gaussian process [49]. It can measure the uncertainty of its prediction, that is approximated with a Gaussian random variable with mean and standard deviation. This uncertainty (i.e. the standard error) is zero, where the high-fidelity function values are already known (i.e. in an already sampled point), whereas it rises with the distance from these known observations.
Hence, while regression models make simplistic assumptions about the independence of the errors, in order to focus on the regression coefficients estimation and to completely describe the function, interpolation models prefer to simplify the assumptions about the regressors and their coefficients. They consider the regressors just as constants, but give greater importance to the evaluation of the errors correlation, which allows to describe just the typical behaviour of the function [47].
Splines are also a set of interpolation functions, i.e. they are piecewise polynomials that approximate functions over large intervals where single polynomial could lose performance. The most commonly used method is that of cubic splines, because it allows to obtain smoother functions that fit the data, reducing the oscillatory behaviour that could result from higher-order degree polynomials [50].
Radial basis functions belongs to this category too [51, 32, 34]. They belong to a class of functions whose response changes monotonically with the distance from a central point [45]. This type of approximation employs the combination of K basis functions ϕ as:

$$\hat{f}(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{\phi} = \sum_{i=1}^{K} w_i \phi(\|\boldsymbol{x} - \boldsymbol{c}^{(i)}\|) \tag{2.4}$$

where $\boldsymbol{w}$ is the vector that contains model parameters and that can be evaluated as for regression parameters (Equation 2.2), $\boldsymbol{c}$ is the vector of the known basis function centres. Therefore, the functions ϕ are computed considering the Euclidean distances between the prediction site $\boldsymbol{x}$ and the centres $\boldsymbol{c}^{(i)}$ [32]. Typical choices for the basis functions are: $\phi(r) = r$ (linear), $\phi(r) = r^3$ (cubic), or $\phi(r) = r^2 ln(r)$ (thin plate spline). Adopting these functions as fixed bases, reduces the number of weights to determine, to just one. They are used for higher-dimensional problems than that in which it is preferable to use polynomial models, but for simple landscape and cheap analysis [32]. On the other hand, using more complex parametric basis functions, allows to achieve more flexibility, even if it leads to parameters harder to estimate. Examples of these basis functions are: $\phi(r) = \exp(-r^2/2\sigma^2)$ (Gaussian), $\phi(r) = (r^2+\sigma^2)^{1/2}$ (multi-quadric) or $\phi(r) = (r^2+\sigma^2)^{-1/2}$ (inverse multi-quadric) [34].
Also Kriging is part of basis function models, but it differs from a spline interpolation model, because the latter uses basis functions prefixed in advance, whereas the first, can tune them, with respect to the data (changing θ_h, p_h parameters), so it is more accurate but requires lengthier time [47]. Another advantage of Kriging, with respect to other basis functions, is the possibility to also compute the uncertainty of the prediction it makes. Therefore, Kriging and parametric basis functions allow to reach greater accuracy, but in low-dimensional spaces, because of the resources expense needed to train the model [32].
The accuracy of the approximation achieved for reproducing training data, lies on

the best choice of $\boldsymbol{w}$, while an adequate estimation of the new parameters that the particular basis function brings along (e.g. σ for Gaussian basis functions), affects the generalized model error.

2.3 Multifidelity surrogates

Combining models with varying degrees of accuracy (*fidelity*), can result in a satisfying representation of the reality, without exceeding the available resources, thanks to a further reduction of the amount of high-fidelity evaluations required. A multifidelity surrogate, indeed, is built taking into account the data available from a narrow number of high-fidelity model evaluations. For what concerns low-fidelity models, if they are cheap enough, building a surrogate may not be necessary. The advantage achieved by using multifidelity methods is that multifidelity problems can be handled keeping the system bias with the use of low-fidelity approaches, and refining key performances (unbiased) with the call of high-fidelity models. The challenge lies into the comprehension and implementation of the best strategy to combine different fidelities.
According to Peherstorfer et al. [29], the management strategies (i.e. decide which model to evaluate and when) to combine more fidelities without losing accuracy, are distinguished in three types: *fusion, adaptation* and *filtering*. Surrogates belong to the fusion category. Fusion approach, merges low- and high-fidelity model outputs. Cokriging, which is a multiresponse extension to kriging, is part of these techniques. Response surfaces and kriging surrogates are the most used methods belonging to the class of surrogate models. Therefore, data-fit models and projection-based models, previously described in section 2, can be included in this category of modeling techniques [30]. For what concerns the adaptation strategy, it updates and enriches the low-fidelity model with information gained from the high-fidelity model, while the computation goes on and more reliable high-fidelity data become available. An example is the EGO algorithm for efficient global optimization, where the adaptation process is applied to a kriging model in each iteration of the optimization. Other examples of applications are: the work of Kennedy and O'Hagan [52], where the correction (calibration) of low-fidelity model outputs is based on the employment of Gaussian process models, which have better performances in predicting the output of the high-fidelity model; adopting the strategy of comparing the low-fidelity outputs, with the high-fidelity model updates, in order to define the correction to apply to low-fidelity model [53, 54]. Filtering technique tends to invoke the high-fidelity model only when indicated by a low-fidelity filter: for example, it can be called when the low-fidelity results particularly inaccurate. This strategy can be adopted in a stochastic approach in order to choose, with the low-fidelity model, candidate sample points at which the high-fidelity model has to be evaluated.
However, there are alternative methods that hierarchically combine different fidelity levels through adaptive sampling, without explicitly merging fidelities in a unique surrogate. Those are also referred to as *multifidelity hierarchical methods* and frequently combine the data from the source models through adaptation and/or filtering approaches [29, 30]. It is important to notice that, even if multifidelity methods have been advancing considerably for more than two decades, some important challenges still remain. Almost all the methods are based on the assumption that the high-fidelity model represents the 'truth', whereas it is an approximation of reality.

The relationships assumed by multifidelity methods over different fidelity models can be underestimated and excessively simplified with respect to the reality. Therefore, techniques for understanding model adequacy, that is, for quantifying the difference between experimental data and model outcomes, should encompass both expert opinion and statistical criteria for a better overview of the problem [55]. Another important challenge is to try to implement methods which can combine a wider range of information sources, such as experiments, lookup tables, computational models and experts' opinions [29]. Studies about the fusion of different information sources (Multi-Information Source Optimization, MISO) are those study made by Poloczek et al. [56] and Ghoreishi et al. [57]. They provide multifidelity Bayesian methods to handle the discrepancies of different information source bias.

2.4 Multifidelity Optimization

The literature [29] proposes to classify the problems to be solved using multifidelity methods, into three categories: optimization, uncertainty quantification and statistical inference problems. Hereafter, it will be a focus on optimization applications. Multifidelity optimization is an iterative process which aims to find the optimum of the high-fidelity function. Typically, the first step involves the realization of a surrogate model, then the method leverages the surrogate for searching the optimum. Optimization is distinguished in global and local approaches. The first searches the optimum over the entire domain, whereas the latter narrows the research locally, around the current best design point. Global multifidelity optimization has the advantage of overriding the evaluation of the gradient of the high-fidelity function. This is useful when the gradient cannot be evaluated accurately (e.g. it may be affected by noise). It can be the case of a black-box function. However, global search requires a higher amount of high-fidelity function evaluations than local optimization methods.

2.4.1 Global Multifidelity Optimization

Global multifidelity optimization methods search for a minimizer taking into account an iterative upgrade and enhancement of low-fidelity models. Typically, they do not employ gradient-based optimizers, whereas they prefer to build a cheap global surrogate of the high-fidelity model. Then, this surrogate replaces the original model and is adopted for optimization purposes. The resulting optimal solution will be also a minimizer of the high-fidelity model, to the extent that the level of accuracy has been achieved by low-fidelity model. It is important to select appropriately the assumptions on which the surrogate model lies, because if the problem cannot satisfy them, the accuracy of the surrogate may suffer [30].
Cokriging surrogate models are used in global multifidelity optimization processes. Updating iteratively Cokriging models in these contexts, allows to find the global optimum more quickly and accurately than other methods because, from the fusion of different fidelity model, the accuracy of the surrogate model increases. However, it may become more expensive with increasing dimensionality [24, 58].
Weakness of these optimization methods is the necessity to make an adequate trade-off between exploitation and exploration. Promoting only exploration leads to spread a large amount of points over the domain of design, improving the accuracy of the

model, but spending resources in an inefficient way; on the other hand, preferring only an exploitation approach, that tries to search in an almost good point, leads to the possibility to not catch the optimum location if an appropriate exploration has not been set before. However, this problem will be widely discussed in Chapter 3. It is to highlight that when using surrogate-based optimization, ill conditioning of the correlation matrices may arise, especially when the algorithm is close to convergence, because sampled points are close together: an appropriate convergence criteria needs to be set to override the problem [58].
Also EGO belongs to this category of optimization methods. It is frequently chosen to model the low-fidelity model with a Kriging model. After fitting the response surface, maximizing the expected improvement (i.e. maximizing the difference between the best current evaluation of the high-fidelity model and the mean of the Gaussian random variable), leads to select the next point to query with the high-fidelity model. The Kriging model is updated with information derived from this new evaluation, and the optimization process restarts iteratively [59]. The EGO algorithm was extended to a multifidelity context, thanks to Kennedy and O'Hagan. They fitted Gaussian process models to lower-fidelity models, in order to best predict the output of high-fidelity ones [52]. Therefore, several Bayesian-based methods arise from EGO approach with the purpose of driving an adaptive sampling in a goal-driven perspective.

2.4.2 Local Multifidelity Optimization

Local multifidelity optimization, typically uses trust-region methods. They implement quadratic approximations of the high fidelity model at the center of the trust region. This approximation is computed by evaluating gradient and Hessian of the high-fidelity model. The quality of the approximation determines the size of the trust region and so, the allowable step length: the optimization can be done only within the portion of the domain in which the local approximation is valid. To accelerate a provably convergent local optimization method, response surfaces result useful, because require less high-fidelity evaluations. It is preferred to implement response surfaces which interpolate the data available with a linear combination of basis functions that can be fixed or can have parameters to be tuned (e.g. Kriging). An example of local multifidelity optimization method that chooses surface responses is that proposed in [60]. They use Kriging response surface to predict the function value at points closed to that currently sampled, reducing substantially the number of high-fidelity function evaluations. Then, also Alexandrov et al. [61] adopt response surfaces. They build surrogate models by improving low-fidelity models, thanks to information about functions observations and their gradients that derive from the high fidelity. Hence, response surfaces result more accurate in finding the next point to evaluate, than minimizing Taylor-series approximation within the trust region. Their greater accuracy allow to take longer steps, making the algorithm faster. Forcing the surface and the high-fidelity function to have equal gradient at the current iteration in the trust region, is an essential requirement for convergence. The selection of the next point to query is made optimizing the surface within a trust region. If the point just selected, does not lead to the objective function reduction, then the trust region is updated and narrowed. In this way, the method can converge to a critical point of the high-fidelity function, which not necessarily

coincides with a global optimum or a local one, but can be a saddle point or a point in which the function results flatter. It strongly depends from the initial set of high-fidelity samples adopted for all these methods [59, 29]. When gradients cannot be easily evaluated or result inaccurate (e.g. in case of black-box functions), gradient-free multifidelity trust-region approaches can be adopted. For example, March and Willcox [62] have developed a method that allows to achieve a first-order optimality using information from iteratively calibrated low-fidelity function to high-fidelity one. This method is built referring either to Conn et al. [63, 64] or to Wild et al. [51, 65, 66]. The first proved that using polynomial interpolation models, can lead to a fully linear model (which assures smoothness of surrogate models of high-fidelity function) without requiring to estimate any high-fidelity gradient; the latter demonstrated that a radial basis function interpolation is also efficiently adaptable to Conn's method. Therefore, March and Willcox, generalize this method to a multifidelity optimization framework.
Comparing gradient-free and gradient-based methods, shows that the first starts rapidly and then, slows down in proximity of the best solution; whereas the latter, assumes an opposite behaviour (as Hessian becomes more accurate).

Chapter 3

Active learning via Bayesian Optimization

It is possible to choose between two methods, for evaluating multifidelity surrogates parameters: deterministic or non-deterministic methods. First takes advantage of assuming known basis functions, whose coefficients are found by minimizing discrepancy between the data and the function [67, 68]. The latter, assumes that the function or its coefficients are uncertain [69]. This uncertainty requires to be quantified and minimized via statistical inference methods, like Bayesian framework, which results less expensive than other techniques, such as Monte Carlo simulations. Non-deterministic methods result to be more accurate than deterministic ones [70, 71].
This chapter, drawn from [49], will focus on Bayesian Optimization (BO), that are machine learning methods for optimizing black-box functions based on stochastic processes. The goal of these methods is to find the global optimum of an objective function, whose behaviour is not known a priori, reducing considerably the number of expensive or unfeasible evaluations required. They inscribe into the category of SBO, because they model the unknown function with a surrogate model, whose characteristics are computed via Gaussian processes. Furthermore, they use an acquisition function for deciding where to sample next the objective function. They implement the active learning schemes to leverage the data dynamically collected during the optimization process, thus providing a technique for finding the most relevant regions of the domain.
In Chapter 2, we have assessed how to build a surrogate. The following step, in a Bayesian optimization process, focuses on the research of *figures of merit* that try to balance local and global search for where to take next evaluations [47]. This means adopting an acquisition function that makes a trade-off between a global *exploration* and a local *exploitation* of the surrogate model, in order to minimize the number of objective function evaluations. The term 'exploration' means adding input points to query, in order to gain information where the objective function is really uncertain, whereas 'exploitation' means choosing points to guide a further improvement of the GP belief at an already good point. This is the reason why there is interest not only for the maximization of the acquisition function, but also for learning its overall behaviour over the entire search domain. Therefore, the acquisition function guides the surrogate model along the search path of promising candidates. Typically, acquisition functions are defined in order that, when they assume high values, they correspond to potentially high values of the objective function, whether because the

prediction is high, the uncertainty is great, or both. Then, the evaluation of the objective function is computed at that selected point $x_{t+1} \in \mathcal{A}$, and the model is updated with the corresponding function value: the process repeats. The optimization ends after a maximum number of iterations, or a stopping criterion, such as spending all the available budget, is met. Figure 3.1, easily explains the concept

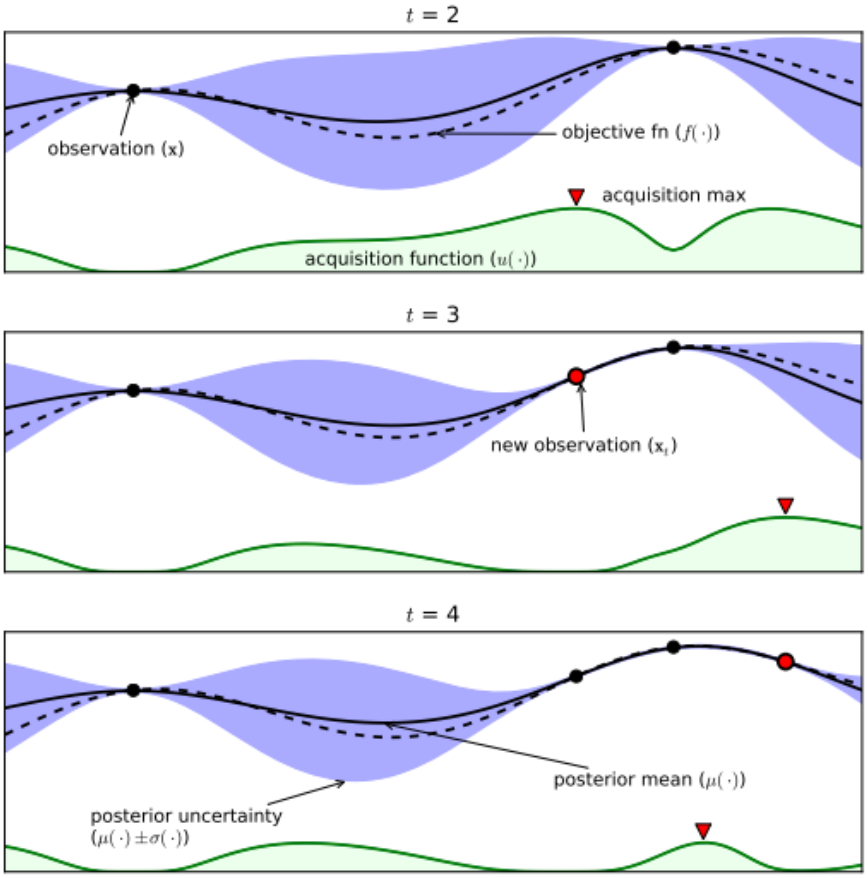


Figure 3.1: Example of BO application proposed by Brochu et al. [49]

of BO applied to a 1D design problem. For the first iteration, two initial samples are selected. At each iteration, the acquisition function, in green, is maximized to determine where next to sample the objective function. This function takes into account the mean and variance of the predictions over the space of design. It is high where the GP predicts a high objective (exploitation) and where its uncertainty is high (exploration). The next evaluation of the objective function will be in correspondence of the *argmax* of the acquisition function. After the addition of this new point to the initial sampling, the GP is uploaded. The area on the left of the figure, remains unsampled because of the high uncertainty that cannot offer a useful information gain about the optimum. In this way, it is demonstrated how, adopting the appropriate acquisition function, helps to minimize the number of objective function evaluations. It is called 'Bayesian' because it uses the 'Bayes' theorem', which states that the *posterior* probability of a model M, given evidence (or data, or observations) E, is proportional to the *likelihood* of E, given M, multiplied by the *prior* probability of M as follows [49]:

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

where the prior distribution represents what (i.e. which properties) it is supposed to know (before looking at the observations) about the space of possible objective functions, in order to make some functions more plausible than others. Therefore, assuming, for example, that the prior belief is a very smooth objective function, data with high variance should be considered less than that barely deviate from the mean [49]. On the basis of what has just been explained, it is possible to find out the posterior distribution as:

$$P(f|\mathcal{D}_{1:t}) \propto P(\mathcal{D}_{1:t}|f)P(f)$$

The posterior captures the updated beliefs about the unknown objective function. This step can be seen as an estimation of the objective function with the use of a *surrogate function* (*response surface*) with the posterior mean function of a GP. As the observations are accumulated $\mathcal{D}_{1:t} = \{\boldsymbol{x}_{1:t}, f(\boldsymbol{x}_{1:t})\}$, the prior distribution $P(f)$ is combined with the likelihood function $P(\mathcal{D}_{1:t}|f)$ to produce the posterior distribution $P(f|\mathcal{D}_{1:t}) \propto P(\mathcal{D}_{1:t}|f)P(f)$.
Summarizing, BO uses the prior and evidence to define a posterior distribution over the space of the functions. Prior can describe the attributes of the unknown objective function, such as smoothness or the most likely location of the maximum. Then, the optimizing step, allows to decide where to sample next the surrogate model. The choice is guided by the utility function $u(\cdot)$ adopted. It is selected in order to be easy to evaluate, and with the aim of optimizing its expectation with respect to the posterior distribution of the objective function. Following the implementation

Algorithm 1 Bayesian Optimization

1: **for** $t = 1, 2, \ldots$ **do**
2: Find $\mathbf{x}_t$ by optimizing the acquisition function over the GP: $\mathbf{x}_t = \text{argmax}_{\mathbf{x}}\, u(\mathbf{x}|\mathcal{D}_{1:t-1})$.
3: Sample the objective function: $y_t = f(\mathbf{x}_t) + \varepsilon_t$.
4: Augment the data $\mathcal{D}_{1:t} = \{\mathcal{D}_{1:t-1}, (\mathbf{x}_t, y_t)\}$ and update the GP.
5: **end for**

Figure 3.2: Bayesian Optimization Algorithm [49].

of a Bayesian Optimization, given by [49], the first assumption is that the problem is concerned as a maximization one: $\boldsymbol{x}^* = argmax_{\boldsymbol{x}} f(\boldsymbol{x})$. Obviously, it can be changed into a minimization problem, transforming the function as : $g(\boldsymbol{x}) = -f(\boldsymbol{x})$. Then, it is assumed that the objective is *Lipschitz-continuous*, i.e. there exists some constant C, such that: for all $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{A}, \|f(\boldsymbol{x}_1) - f(\boldsymbol{x}_2)\| \leq \mathcal{C}\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|$, though C may be unknown. The problem can be made more narrow by defining it as one of *global* optimization, rather than *local* one. Furthermore, another assumption can be done: consider that all the dimensions have bounds, on the search space, that are all axis-aligned. In this way, the search space becomes a hyperrectangle of dimension d. Despite the simplifications, the number of the objective function evaluations f needed for guaranteeing the best observation $f(\boldsymbol{x}^+) \geq f(\boldsymbol{x}^*) - \epsilon$, may be as large as $(C/2\epsilon)^d$ where the Lipschitz constant C, generally, is a large number. The reason for that, lies in the idea of achieving a prefixed accuracy taking, at the same time, into account the possibility of occurrence of the worst possible case. Usually such worst case, quite never occurs in the practice, so the idea of taking care

only of cases that have a non negligible chance to occur, arises. For this purpose, the global optimization problem should be reformulated superimposing, to the class of problems to be considered, a probabilistic structure and an accuracy criterion is set consequently. The Bayes theorem provides the basic tool for adapting the probabilistic structure to the information gained about the problem through function evaluations [72]. The goal is to use evidence and prior knowledge to maximize the posterior at each step, in order that each new evaluation can decrease the distance between the true global optimum and the expected one given by the model.

3.1 Bayesian statistical model: Gaussian Process

3.1.1 Priors

Generally, the prior distribution is based on a Bayesian approach which employs a Gaussian Process Regression (GPR) and needs to satisfy the following conditions, so that the optimization method will converge to the optimum: i) the acquisition function has to be continuous and has to minimize the risk (i.e. the expected deviation from the global minimum at a fixed point $\boldsymbol{x}$); ii) the conditional variance has to converge to zero if and only if the distance to the nearest observation is zero [73] iii) the objective function has to be continuous; iv) the prior has to be homogeneous; v) the optimization has to be independent of the m^{th} differences. However, it is important to emphasize that the priors are probabilistic assumptions about the data, so they have to be set carefully, avoiding to employ priors which are too narrow and that can rule out reasonable explanations of the data [74]. Močkus [73] showed that, with these assumptions, the GP prior is well-suited to the task.
It is known that a Gaussian process is a collection of random variables (in this case, the values of the unknown function $f(\boldsymbol{x})$ at location $\boldsymbol{x}$), that have a joint Gaussian distribution. Hence, a Gaussian Process is used to describe a distribution over functions: just as a Gaussian distribution is a distribution over a random variable, completely expressed by its mean and covariance, a GP is a distribution over functions, completely described by its mean function m, and covariance function k : $f(\boldsymbol{x}) \sim \mathcal{GP}(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}'))$. So a GP returns the mean and variance of a normal distribution over the possible values of f at $\boldsymbol{x}$ (Figure 3.3). It is assumed, for simplicity, that the prior mean is the zero function $m(\boldsymbol{x}) = 0$. With the addition of more datapoints to the posterior distribution, the mean function adjusts itself to pass through these points, in order that the posterior uncertainty would reduce close to the observations.

3.1.2 The covariance function

The key factor that controls the properties of a Gaussian process is the covariance function k. It is also called *kernel*, because it is a function of two arguments that maps a pair of input $\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{X}$ into $\mathbb{R}$. The most widely-used kernel, in these contexts, is the squared exponential function:

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{1}{2}\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2\right). \tag{3.1}$$

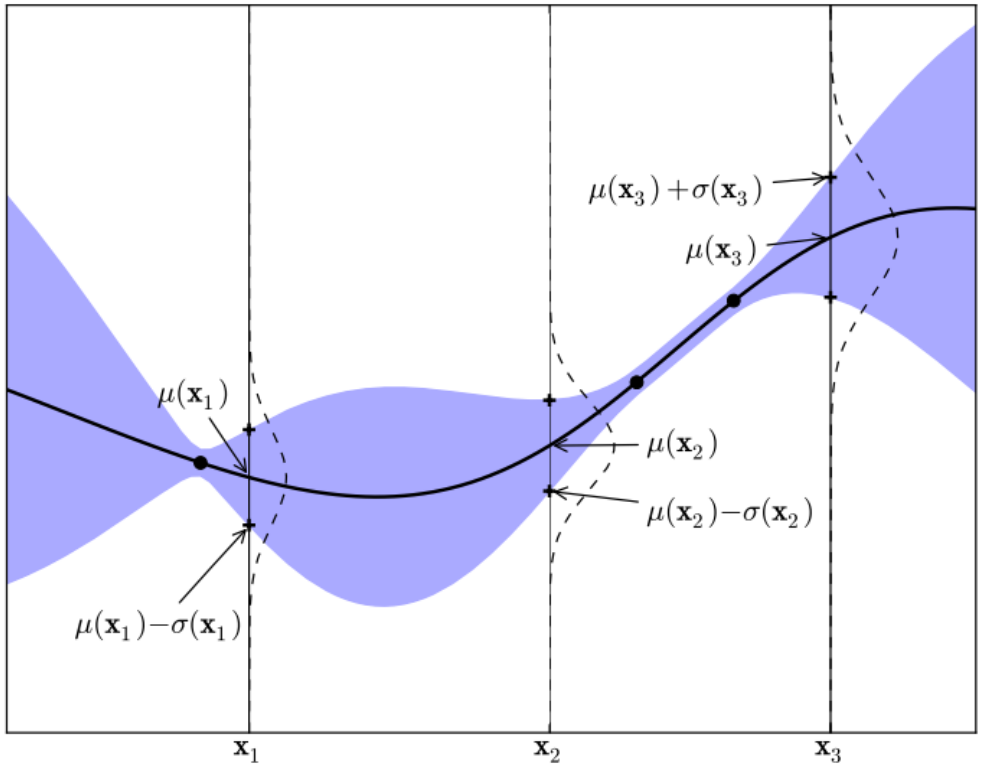


Figure 3.3: 1D Gaussian process with three observations (Brochu et al. [49]). The solid black line is the GP surrogate mean prediction of the objective function and the shaded area represents the mean plus and minus the variance. The superimposed Gaussians correspond to the GP mean and standard deviation $\mu(\cdot)$ and $\sigma(\cdot)$ of prediction at the points $\boldsymbol{x}_{1:3}$.

The function results infinitely differentiable, which means that the GP that adopts it, has mean square derivatives of all orders, making it very smooth [74]. It is possible to recognize, in this formulation, a linear regression model, however, it can also derive from a linear combination of basis functions assuming Gaussian shapes. The choice of a covariance function implies a distribution over functions and is a crucial ingredient in a Gaussian process predictor, as it encodes the assumptions made about the function that has to be learnt: it determines properties of samples drawn from it, such as their smoothness or their characteristic length scale (short length scale means rapid variation of the function). It is a basic assumption that points, with close inputs $\boldsymbol{x}$, are likely to have similar target values y. In the same way, training points, that are near to a test point, should provide information about the prediction at that point. It is just the covariance function that defines nearness or similarity [74]. Therefore, the function tends to the unity, as values get close together, and decreases, tending to zero, as they get further apart into the input space. For this reason, two points that are close together can be expected to have a large influence on each other, whereas distant points have almost none. It represents a necessary condition for convergence under the assumptions of Močkus [73] made before. In the case in which, it is to sample from the the prior, then, it is necessary to choose $\{\boldsymbol{x}_{1:t}\}$, thus that $\mathbf{f}_{1:t} = f(\boldsymbol{x}_{1:t})$ will be the function values assessed at that points. They are drawn according to a multivariate normal distribution $\mathcal{N}(0, \mathbf{K})$ (i.e. a generalization of the univariate normal distribution to two or more variables,

i.e. dimensions), where the kernel matrix is given by:

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

The diagonal values are 1, only in a noise-free environment, such that each point is perfectly correlated with itself. However, in these engineering optimization tasks, the observations $\{\boldsymbol{x}_{1:t}, \boldsymbol{f}_{1:t}\}$ are obtained from previous iterations on an external model. Here, it is assumed to already know these early observations. The next point to be considered, $\boldsymbol{x}_{t+1}$, is chosen with a BO and the value of the function at this point is $f_{t+1} = f(\boldsymbol{x}_{t+1})$, so, by the properties of Gaussian processes, $\mathbf{f}_{1:t}$ and f_{t+1} are jointly Gaussian:

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

where

$$\mathbf{k} = [k(\boldsymbol{x}_{t+1}, \boldsymbol{x}_1) \quad k(\boldsymbol{x}_{t+1}, \boldsymbol{x}_2) \quad \dots \quad k(\boldsymbol{x}_{t+1}, \boldsymbol{x}_t)].$$

Using the Sherman-Morrison-Woodbury formula (also known as *matrix inversion lemma*), the predictive posterior distribution can be expressed as:

$$P(f_{t+1}|\mathcal{D}_{1:t}, \boldsymbol{x}_{t+1}) = \mathcal{N}(\mu_t(\boldsymbol{x}_{t+1}), \sigma_t^2(\boldsymbol{x}_{t+1}))$$

where:

$$\begin{aligned} \mu_t(\boldsymbol{x}_{t+1}) &= \mathbf{k}^T \mathbf{K}^{-1} \mathbf{f}_{1:t} \\ \sigma_t^2(\boldsymbol{x}_{t+1}) &= k(\boldsymbol{x}_{t+1}, \boldsymbol{x}_{t+1}) - \mathbf{k}^T \mathbf{K}^{-1} \mathbf{k}. \end{aligned}$$

A multitude of possible families of covariance functions exists (e.g. squared exponential, polynomial, neural network), each of them typically have a number of free (and sometimes adjustable) parameters, called *hyperparameters*, whose values need to be determined. Choosing a covariance function, for a particular application, thus comprises both selecting hyperparameters within a family, and comparing them across different families [74]. It is assumed that both of these problems will not change inside the same method. Hence, the model selection embraces also the choices made for setting up adequately and accurately the continuous hyperparameters of the covariance function. A Gaussian process model is a non-parametric model, so it may not be trivial to recognize what the parameters of the model are. Among the methods known for selecting a Bayesian model, the cross-validation is considered as an example. It involves splitting a training set into two sets: the first one is used for training, the second one for validation or testing. In this way, the accuracy of the model can be assessed without sampling any points beyond those previously used to fit the model. The method is based on the idea to leave out one observation (evaluated at the point that belongs to the test set), and then predict it back on the basis of the $n - 1$ remaining points [47].
Typically, it is necessary to generalize the squared exponential kernel, described above (3.1), by adding $\boldsymbol{\theta}$, which can be a vector containing all the hyperparameters, if it is contemplated in an anisotropic model; whereas, in an isotropic one, when the

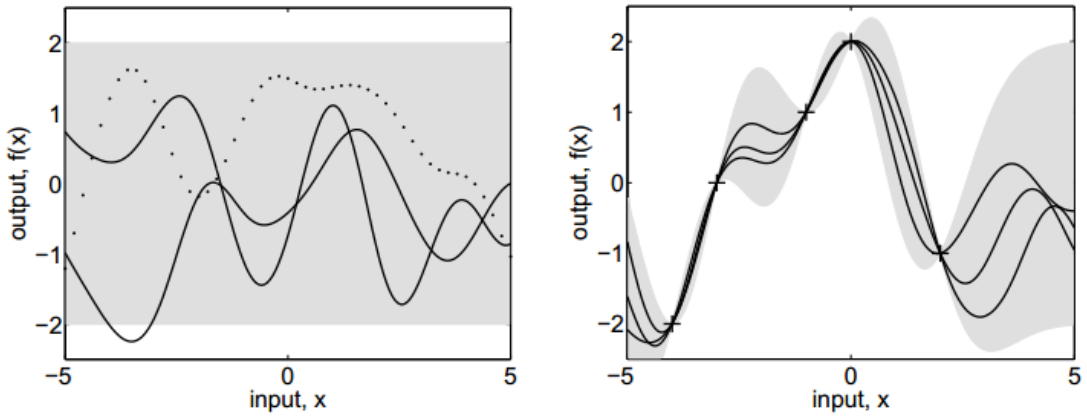


Figure 3.4: Covariance function as a distribution over functions (Williams et al.[74]). Given a number of input points, panel(a) shows three functions drawn from a GP prior. The dots indicate the current y evaluations, whereas the other two functions have been drawn by joining a large number of evaluated points. Panel(b) displays three random functions taken from the posterior, i.e. the prior conditioned on the five noise-free observations indicated as crosses. The shaded area represents the mean plus and minus two times the standard deviation for each input value, for the prior and posterior, respectively.

covariance function is invariant to all rigid motions, θ can be a single hyperparameter, which controls the width of the kernel.
In an isotropic case, it can be expressed as:

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

Whereas, for anisotropic models:

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{x}_j)^T \mathrm{diag}(\boldsymbol{\theta})^{-2}(\boldsymbol{x} - \boldsymbol{x}')\right).$$

where diag($\boldsymbol{\theta}$) is a diagonal matrix with d entries $\boldsymbol{\theta}$ along the diagonal. Intuitively, if a particular component θ_l, of the vector $\boldsymbol{\theta}$, has a small value, the kernel becomes independent of that $l{-}th$ input, effectively removing it from the inference. Generally, these values are learnt by 'seeding' with a few random samples and maximizing the likelihood of the samples given $\boldsymbol{\theta}$ [47, 74].
Another kernel function commonly used for BO, is the Matérn kernel [75, 76]. It has a smoothness parameter ζ that assures greater flexibility in modelling functions:

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

where $\Gamma(\cdot)$ and $H_{\zeta}(\cdot)$ are respectively the Gamma function and the Bessel function of order ζ. As $\zeta \to \infty$, Matérn kernel reduces to the squared exponential one. However, numerous other covariance functions have been examined in the machine learning literature for adapting to different problems [74].

3.1.3 Noisy environment

Even if previously the covariance function has been assumed to be set in a noise-free environment, more realistic modelling situations look different. It cannot be possible to have access to function values themselves $f(\boldsymbol{x})$, but only noisy versions thereof. The noise measurement is assumed, for simplicity, to be Gaussian: $\epsilon \sim \mathcal{N}(0, \sigma^2_{noise})$. If the noise distribution is additive, it can be easily added to the Gaussian distribution $\mathcal{N}(0, \mathbf{K})$ and it is obtained: $y_i = f(\boldsymbol{x}_i) + \epsilon_i$ which shows the noisy observation of the objective function at the $i-th$ sample $\boldsymbol{x}_i$. Since the mean is zero, the kernel $\mathbf{k}$ is replaced as:

$$\mathbf{K} = \begin{bmatrix} k(\boldsymbol{x}_1, \boldsymbol{x}_1) & \dots & k(\boldsymbol{x}_1, \boldsymbol{x}_t) \\ \vdots & \ddots & \vdots \\ k(\boldsymbol{x}_t, \boldsymbol{x}_1) & \dots & k(\boldsymbol{x}_t, \boldsymbol{x}_t) \end{bmatrix} + \sigma^2_{noise} I$$

This yields the predictive distribution:

$$P(y_{t+1}|\mathcal{D}_{1:t}, \boldsymbol{x}_{t+1}) = \mathcal{N}(\mu_t(\boldsymbol{x}_{t+1}), \sigma^2(\boldsymbol{x}_{t+1}) + \sigma^2_{noise}),$$

and the statistics:

$$\mu_t(\boldsymbol{x}_{t+1}) = \mathbf{k}^T [\mathbf{K} + \sigma^2_{noise} I]^{-1} \mathbf{y}_{1:t}$$
$$\sigma^2_t(\boldsymbol{x}_{t+1}) = k(\boldsymbol{x}_{t+1}, \boldsymbol{x}_{t+1}) - \mathbf{k}^T [\mathbf{K} + \sigma^2_{noise} I]^{-1} \mathbf{k}.$$

3.2 Acquisition Functions for BO

Once a statistical model to represent the unknown function has been implemented, it is necessary to choose a sampling strategy for selecting a new query point. The selection technique takes advantage of the posterior distribution to guide the search and usually provides to the maximization of a quantity that measures the amount of information the new current query will provide (its expected utility). Therefore, maximizing the acquisition function means finding the next candidate at which to evaluate the unknown objective function. Hence, the role of the acquisition function is to guide the search for the optimum. Commonly-used acquisition functions are: the Expected Improvement (EI) [47] and the Probability of Improvement (PI), which measure the expected amount by which an improvement can be made over the current best solution; Predictive Entropy Search (PES) [77], Entropy Search (ES) [78], Max-value Entropy Search (MES) which aim to approximate the information gain or the expected reduction in the posterior entropy of the global optimizer of the objective [79, 77, 78]. Hereafter their single-fidelity implementations are showed.

3.2.1 Probability of Improvement

The PI criterion attempts to find the location, where the probability of improving the objective function, based on the current surrogate model, is the highest [59]. Kushner, in 1964, [80] suggested to maximize the probability of improvement over the incumbent $f(\boldsymbol{x}^+)$, where $\boldsymbol{x}^+ = \underset{\boldsymbol{x}_i \in \boldsymbol{x}_{1:t}}{\arg\max} f(\boldsymbol{x}_i)$, so that:

$$PI(\boldsymbol{x}) = P(f(\boldsymbol{x}) \geq f(\boldsymbol{x}^+))$$
$$= \Phi\left(\frac{\mu(\boldsymbol{x}) - f(\boldsymbol{x}^+)}{\sigma(\boldsymbol{x})}\right)$$

where $\Phi(\cdot)$ is the normal cumulative distribution function, $f(\boldsymbol{x}^+)$ is the current optimal solution of the surrogate model, $\mu(\boldsymbol{x})$ and $\sigma(\boldsymbol{x})$ are the mean and standard deviation of a Gaussian distribution. With this formulation, the acquisition function results a pure exploitation. Points that have a high probability of being infinitesimally greater than $f(\boldsymbol{x}^+)$ will be preferred to points that offer major gain but less certainty: it is required to modify the PI formulation in order to set properly a trade-off parameter $\xi \geq 0$.

$$PI(\boldsymbol{x}) = P(f(\boldsymbol{x}) \geq f(\boldsymbol{x}^+) + \xi)$$
$$= \Phi\left(\frac{\mu(\boldsymbol{x}) - f(\boldsymbol{x}^+) - \xi}{\sigma(\boldsymbol{x})}\right)$$

Kushner recommended to start adopting a high ξ in the early stages of the optimization, in order to drive exploration, and then suggested to decrease its value toward zero as the algorithm continues. The exact choice of ξ is left to the user. Studying this approach, Jones [59] noticed that the method is extremely sensitive to the choice of the target. If the desired improvement results too small, the search will tend to be mainly local and will only move on to search globally, after searching exhaustively around the current best point. On the other hand, if the ξ selected is too high, the search will be excessively global, making the algorithm slower.

3.2.2 Expected Improvement

An alternative to this utility function, is the Expected Improvement (EI). It takes into account not only the probability of improvement, but also the magnitude of the improvement a point can potentially yield. The goal is to minimize the expected deviation from the true maximum $f(\boldsymbol{x}^*)$, when choosing a new trial point:

$$\boldsymbol{x}_{t+1} = \arg\min_{\boldsymbol{x}} E(\|f_{t+1}(\boldsymbol{x}) - f(\boldsymbol{x}^*)\| \,|\mathcal{D}_{1:t})$$
$$= \arg\min_{\boldsymbol{x}} \int \|f_{t+1}(\boldsymbol{x}) - f(\boldsymbol{x}^*)\| P(f_{t+1}|\mathcal{D}_{1:t})\, df_{t+1}.$$

This formulation considers one-step ahead choices and, even if it can be easily and recursively extended for as many steps ahead as desired, it can become really expensive. Močkus [73] proposed the alternative of maximizing the expected improvement with respect to $f(\boldsymbol{x}^+)$ as:

$$I(\boldsymbol{x}) = max\{0, f_{t+1}(\boldsymbol{x}) - f(\boldsymbol{x}^+)\}. \tag{3.2}$$

That is, $I(\boldsymbol{x})$ models the uncertainty about the function value at $\boldsymbol{x}$ and is positive, when the prediction is higher than the best value known thus far, otherwise it is set to zero. This expression is a random variable, because $f_{t+1}(\boldsymbol{x})$ is a random variable too. The new query point is found by maximizing the expected improvement:

$$\boldsymbol{x} = \arg\max_{\boldsymbol{x}} E\left(max\{0, f_{t+1}(\boldsymbol{x}) - f(\boldsymbol{x}^+)\} \,|\mathcal{D}_t\right)$$

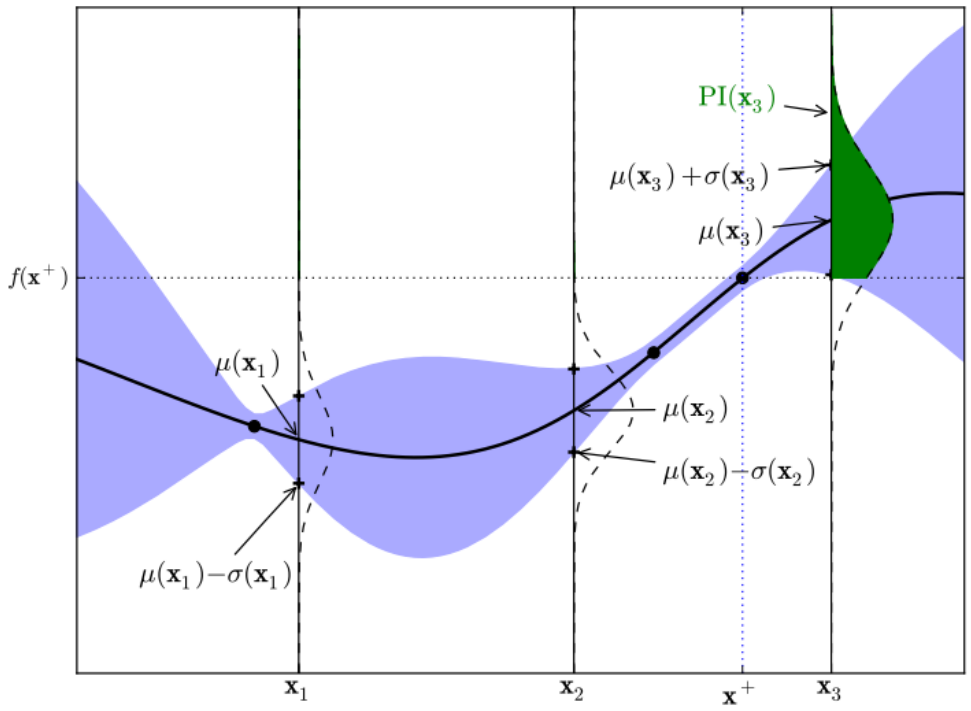


Figure 3.5: Probability of Improvement depiction [49].
The figure shows the region of probable improvement. The maximum observations is at $\boldsymbol{x}^+$, the green area above the dashed line can be seen as a measure of improvement $I(\boldsymbol{x})$. The model predicts a major possibility of improvement by observing at $\boldsymbol{x}_3$ rather than at $\boldsymbol{x}_1$ or $\boldsymbol{x}_2$.

The likelihood of improvement $I(\boldsymbol{x})$, on a normal posterior distribution, characterized by $\mu(\boldsymbol{x})$ and $\sigma^2(\boldsymbol{x})$, can be computed from the normal density function:

$$\frac{1}{\sqrt{2\pi}\sigma(\boldsymbol{x})}\exp\left(-\frac{\left(\mu(\boldsymbol{x})-f(\boldsymbol{x}^+)-I\right)^2}{2\sigma^2(\boldsymbol{x})}\right). \tag{3.3}$$

Therefore, the expectation is calculated by integrating, by parts, over this function:

$$\begin{aligned} E(I) &= \int_{I=0}^{I=\infty} I\frac{1}{\sqrt{2\pi}\sigma(\boldsymbol{x})}\exp\left(-\frac{\left(\mu(\boldsymbol{x})-f(\boldsymbol{x}^+)-I\right)^2}{2\sigma^2(\boldsymbol{x})}\right)dI \\ &= \sigma(\boldsymbol{x})\left[\frac{\mu(\boldsymbol{x})-f(\boldsymbol{x}^+)}{\sigma(\boldsymbol{x})}\Phi\left(\frac{\mu(\boldsymbol{x})-f(\boldsymbol{x}^+)}{\sigma(\boldsymbol{x})}\right)+\phi\left(\frac{\mu(\boldsymbol{x})-f(\boldsymbol{x}^+)}{\sigma(\boldsymbol{x})}\right)\right] \end{aligned}$$

In closed form, the expected improvement becomes:

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

$$Z = \frac{\mu(\boldsymbol{x})-f(\boldsymbol{x}^+)}{\sigma(\boldsymbol{x})}$$

where $\phi(\cdot)$ and $\Phi(\cdot)$ represent the Probability Density Function (PDF) and the Cumulative Density Function (CDF) of the standard normal distribution.
In a noisy environment, the expression of PI and EI acquisition functions has to

be changed. Instead of using the best observation as incumbent, it is used the distribution at the sampled points, and then it is selected the point with the highest expected value:

$$\mu^+ = \arg\max_{\boldsymbol{x}_i \in \boldsymbol{x}_{i:t}} \mu(\boldsymbol{x}_i).$$

This formulation helps to avoid attempting to maximize probability or expected improvement over an unreliable sample.

3.2.3 Predictive Entropy Search

Entropy search methods find a query point that maximizes the information about the location of the target maximizer $\boldsymbol{x}_* = \arg\max_{x \in \mathcal{X}} f(\boldsymbol{x})$ whose value $y_* = f(\boldsymbol{x}_*)$ represents the global maximum of the function f, which is a function sampled from $\mathrm{GP}(\mu, k)$. PES adopts the following acquisition function that uses a negative differential entropy of $p(\boldsymbol{x}_*|\mathcal{D}_t)$ to characterize the uncertainty about $\boldsymbol{x}_*$ [81]:

$$\begin{aligned}\alpha_t(x) &= I(\{\boldsymbol{x}, y\}; \boldsymbol{x}_*|\mathcal{D}_t) \\ &= H(p(y \mid \mathcal{D}_t, \boldsymbol{x})) - E[H(p(y \mid \mathcal{D}_t, \boldsymbol{x}, \boldsymbol{x}_*))].\end{aligned}$$

Unfortunately, whether $p(x_*|\mathcal{D}_t)$ or its entropy are not analytically feasible and need to be approximated with expensive computations. Furthermore, the optimum may not be unique, making the evaluation more complex.

3.2.4 Max-value Entropy Search

To avoid the problems just exposed about the PES, Wang and Jegelka [81] propose to measure the information about the maximum value $y_* = f(\boldsymbol{x}_*)$, making the search path one-dimensional. The acquisition function becomes the gain in mutual information between the maximum y_* and the next candidate point, which can be approximated analytically, by evaluating the entropy of the predictive distribution:

$$\begin{aligned}\alpha_t(x) &= I(\{\boldsymbol{x}, y\}; y_* \mid \mathcal{D}_t) \\ &= H(p(y \mid \mathcal{D}_t, \boldsymbol{x})) - E[H(p(y \mid \mathcal{D}_t, \boldsymbol{x}, y_*))] \\ &\approx \frac{1}{K} \sum_{y_* \in Y_*} \left[\frac{\gamma_{y_*}(\boldsymbol{x}) \psi(\gamma_{y_*}(\boldsymbol{x}))}{2\Psi(\gamma_{y_*}(\boldsymbol{x}))} - \log(\Psi(\gamma_{y_*}(\boldsymbol{x}))) \right]\end{aligned} \tag{3.4}$$

where ψ is the PDF, Ψ is the CDF of a normal distribution and $\gamma_{y*}(\boldsymbol{x}) = \frac{y_* - \mu_t(x)}{\sigma_t(x)}$. The expectation, here, is over $p(y_*|\mathcal{D}_t)$, which is much easier to compute with respect to ES and PES, and is approximated using Monte Carlo estimation, by sampling a set of K function maxima. Furthermore, the probability on the left side of the Eq.(3.4), $p(y|\mathcal{D}_t, \boldsymbol{x})$ is a Gaussian distribution with mean and variance that are, respectively, $\mu_t(\boldsymbol{x})$ and $k_t(\boldsymbol{x}, \boldsymbol{x})$. Whereas, the probability on the right side, $p(y|\mathcal{D}_t, \boldsymbol{x}, y_*)$ is a truncated distribution, so that, given y_*, the distribution of y needs to satisfy $y < y_*$. To sample y_*, Wang and Jegelka show two strategies: sampling from an approximation via a Gumbel distribution; or sampling by maximizing functions from the posterior Gaussian distribution. Adopting the Gumbel Sampling approach, means to sample from the approximation $\hat{p}(y^*|\mathcal{D}_n)$ via its CDF

$\widehat{Pr}[y_* < z] = \prod_{x \in \hat{X}} \Psi(\gamma_z(\boldsymbol{x}))$, where $\hat{X}$ is a discrete subset of representative points that replaces, for simplicity, the continuous set X. This means that it is necessary to sample r uniformly from [0,1] and find z, such that $Pr[y_* < z] = r$. This search can take time, so a more time-strategic computation is showed, by approximating the CDF by a Gumbel distribution: $\widehat{Pr}[y_* < z] \approx \mathcal{G}(a, b) = e^{-e^{-\frac{x-a}{b}}}$. This choice is motivated by Fisher-Tippett-Gnedenko theorem [82], which states that the maximum of a set of i.i.d. (Independent and Identically Distributed random variables) Gaussian variables is asymptotically described by a Gumbel distribution. Sampling from the Gumbel distribution is made via the Gumbel quantile function, and the appropriate set of parameters a and b is obtained by percentile matching. An alternative method, is that of sampling a set of function $\tilde{f}$, drawn from the GP posterior, and then maximize them in order to get samples of $y_* = max_{x \in X} \tilde{f}(\boldsymbol{x})$. However, further information can be found in [81, 77].

Chapter 4

Active learning via multiple sources: Multifidelity Bayesian Optimization

More efficiency, with respect to the Bayesian Optimization just showed, can be achieved in a multifidelity framework (Multifidelity Bayesian Optimization, MFBO), where having cheaper level of approximations of the objective function, provides less costs and more useful information to assist and accelerate the search of optimum.
In a multifidelity approach, the optimization process has to choose either the next iterates, or which fidelity model to adopt for that evaluation, simultaneously. Therefore, an adjustment to multiple fidelities is required in the definition of Gaussian Process Regression (GPR) prior seen in section 3.1.1. Consider that $y_x^{(1)}, \dots, y_x^{(M)}$ are the observations at $x \in \mathcal{X}$, where $\mathcal{X}$ is the space of design and M are the different fidelities expressed in ascending order for $m = 1, \dots, M$. Each observation is modeled as $y_x^{(m)} = f_x^{(m)} + \epsilon$, where ϵ represents the measurement noise, which is assumed to have equal distribution over the different fidelities, $\epsilon \sim \mathcal{N}(0, \sigma^2_{noise})$. The training data set $D_n = \{(x_i, y_{x_i}^{(m_i)}, m_i)\}_{i \in [n]}$ consists of n inputs x_i, fidelity m_i and output $y_{x_i}^{(m_i)}$, which is the observation at that n-th point. The querying cost is $\lambda^{(m)}$, where $\lambda^{(1)} \leq \lambda^{(2)} \dots \leq \lambda^{(M)}$. Following an autoregressive scheme [52], the lowest fidelity function is modeled with a GP prior $f^{(1)} \sim GP(0, k_1(\boldsymbol{x}, \boldsymbol{x'}))$, where $k_1 : \mathbb{R}^{d \times d} \to \mathbb{R}$ is the kernel function. The higher fidelities are defined recursively, as:

$$f^{(m)}(\boldsymbol{x}) = \rho f^{(m-1)}(\boldsymbol{x}) + \delta^{(m)}(\boldsymbol{x}) \qquad m = 2, \dots, M$$

where ρ is a constant term used for scaling the contribution of the previous fidelity to the following one, and $\delta^{(m)} \sim GP(0, k_m(\boldsymbol{x}, \boldsymbol{x\prime}))$ models the bias between fidelities. According to the autoregressive formulation:

$$cov[f^{(m)}(\boldsymbol{x}), f^{(m-1)}(\boldsymbol{x'}) | f^{(m-1)}(\boldsymbol{x})] = 0 \qquad \forall \boldsymbol{x} \neq \boldsymbol{x'}$$

This can be interpreted as a Markov property, so that nothing more can be learnt about the current evaluation $f^{(m)}(\boldsymbol{x})$ from any other previous model evaluation $f^{(m-1)}(\boldsymbol{x'})$, for $\boldsymbol{x} \neq \boldsymbol{x'}$. Therefore, the expensive simulation is assumed to be correct and any inaccuracies lie in the cheaper models.
Fitting a GPR to a multifidelity context, the so called MF-GPR, allows to define a kernel matrix $\boldsymbol{K} \in \mathbb{R}^{n \times n}$ in which $\boldsymbol{K}(i, j) = k((\boldsymbol{x}_i, m_i), (\boldsymbol{x}_j, m_j))$ for a pair of training

points $(x_i, y_{x_i}^{(m_i)}, m_i)$ and $(x_j, y_{x_j}^{(m_j)}, m_j)$ and all the fidelities are unified into a GPR model, where the predictive distribution is defined by predictive mean and variance [83]:

$$\mu_x^{(m)} = \boldsymbol{k_n^{(m)}}(\boldsymbol{x})^T[\boldsymbol{K} + \sigma_{noise}^2 I]^{-1}\boldsymbol{y}$$

$$\sigma_x^{2(m)} = k((\boldsymbol{x}, m), (\boldsymbol{x}, m)) - \boldsymbol{k_n^{(m)}}(\boldsymbol{x})^T[\boldsymbol{K} + \sigma_{noise}^2 I]^{-1}\boldsymbol{k_n^{(m)}}(\boldsymbol{x}).$$

where:

$$\boldsymbol{I} = \text{is the identity matrix,}$$

$$\boldsymbol{y} \coloneqq (y_{x_1}^{(m_1)}, \ldots, y_{x_n}^{(m_n)})^T,$$

$$\boldsymbol{k_n^{(m)}}(\boldsymbol{x}) \coloneqq (k((\boldsymbol{x}, m), (\boldsymbol{x_1}, m_1)), \ldots, k((\boldsymbol{x}, m), (\boldsymbol{x_n}, m_n)))^T.$$

4.1 Analysis of Multifidelity acquisition functions

Hereafter it is presented an extension, to the multifidelity framework, of all the single-fidelity acquisition functions just debated in Chapter 3.

4.1.1 Multi-Fidelity Expected Improvement

In the case of multiple fidelities, Huang et al. [84], in 2006, proposed to extend the Expected Improvement, implemented by Jones [47] in a single fidelity context, as follows:

$$\text{MFEI}(\boldsymbol{x}, m) \equiv E\left[max\left(f^{(M)}(\boldsymbol{x}) - f^{(M)}(\boldsymbol{x}^+), 0\right)\right] \cdot \alpha_1(\boldsymbol{x}, m) \cdot \alpha_2(\boldsymbol{x}, m) \cdot \alpha_3(m) \quad (4.1)$$

where

$$E\left[max\left(f^{(M)}(\boldsymbol{x}) - f^{(M)}(\boldsymbol{x}^+), 0\right)\right] \quad (4.2)$$

is the expected improvement evaluated at the highest fidelity function (i.e. the current best prediction response of the high-fidelity objective function, $f^{(M)}(\boldsymbol{x}^+)$) and $f^{(M)}(\boldsymbol{x})$ denotes the posterior mean; whereas, the other terms appearing in the formula, are utility functions:

$$\alpha_1(\boldsymbol{x}, m) = \text{corr}\left[f^{(m)}(\boldsymbol{x}), f^{(M)}(\boldsymbol{x})\right], \text{ (where corr stands for correlation)}$$

$$\alpha_2(\boldsymbol{x}, m) = 1 - \frac{\sigma_\epsilon}{\sqrt{\sigma^{2(m)}(\boldsymbol{x}) + \sigma_\epsilon^2}}, \qquad \text{where } \sigma^{2(m)}(\boldsymbol{x}) = cov[f^{(m)}(\boldsymbol{x}), f^{(m)}(\boldsymbol{x})]$$

$$\alpha_3(m) = \frac{\lambda_M}{\lambda_m}$$

The term $\alpha_1(\boldsymbol{x}, m)$ takes into account the reduction in reward when a lower-fidelity evaluation is used. When $m = M$, the term is equal to one; whereas, it should be zero when already exists an evaluation at $(\boldsymbol{x}, m)$ and there is not random error, because replicates bring no additional information. It represents the correlation between the posterior estimate of systems m and M at the same point $\boldsymbol{x}$. It can be

seen as the fraction of uncertainty that can be deleted from the high-fidelity system M, once m is known.
The term $\alpha_2(\boldsymbol{x}, m)$ helps to adjust EI, when noise is found in system m outputs. It allows to reduce the addition of replicates as the prediction becomes more accurate, therefore this factor is equal to one when the variance of the random error is zero.
The term $\alpha_3(m)$ identifies the ratio between the cost-per-evaluation on the higher fidelity system with respect to that on th m-th system . With equal expected gains, the algorithm will prefer to choose a cheaper datapoint to a more expensive one.
To summarize, the presence of the first term is necessary to avoid choosing always the cheaper low-fidelity system; the second factor allows to eliminate unnecessary replicates that could arise when random errors exist; excluding the last term could bring to always select the highest-fidelity. The expectation can be analytically computed by integrating over the distribution $f^{(M)}(\boldsymbol{x})$, with $f^{(M)}(\boldsymbol{x}^+)$ as a fixed value:

$$E\left[max\left(f^{(M)}(\boldsymbol{x}) - f^{(M)}(\boldsymbol{x}^+), 0\right)\right] =$$

$$(f^{(M)}(\boldsymbol{x}) - f^{(M)}(\boldsymbol{x}^+))\Phi\left(\frac{f^{(M)}(\boldsymbol{x}) - f^{(M)}(\boldsymbol{x}^+)}{\sigma^{(m)}(\boldsymbol{x})}\right) + \sigma^{(m)}(\boldsymbol{x})\phi\left(\frac{f^{(M)}(\boldsymbol{x}) - f^{(M)}(\boldsymbol{x}^+)}{\sigma^{(m)}(\boldsymbol{x})}\right)$$

4.1.2 Multi-Fidelity Probability of Improvement

MFPI is here implemented, according to [85] that developed this method in 2019. Always is a maximization problem, it is formulated as:

$$\text{MFPI}(\boldsymbol{x}, m) = PI(\boldsymbol{x}) \times Corr(\boldsymbol{x}, m) \times CR(m) \times \eta(\boldsymbol{x}, m) \tag{4.3}$$

where

$$PI(\boldsymbol{x}) = \Phi\left(\frac{f^{(M)}(\boldsymbol{x}) - f^{(M)}(\boldsymbol{x}^+)}{\sigma^{(M)}(\boldsymbol{x})}\right) \tag{4.4}$$

$$Corr(\boldsymbol{x}, m) = Corr\left[f^{(M)}(\boldsymbol{x}), f^{(m)}(\boldsymbol{x})\right] \tag{4.5}$$

$$CR(m) = \frac{\lambda_M}{\lambda_m} \tag{4.6}$$

$$\eta(\boldsymbol{x}, m) = \prod_{i=1}^{n}\left[1 - R\left(\boldsymbol{x}, \boldsymbol{x}_i^{(m)}\right)\right]. \tag{4.7}$$

The term $Corr(\boldsymbol{x}, m)$ represents the posterior correlation between m-th and M-th fidelity models evaluated at the same input point. It reaches, indeed, its maximal value of one, when a high-fidelity sample is added to the setpoint, so that the PI cannot be reduced at the current iteration: it quantifies the penalty connected to the selection of the lower-fidelity function.
The term $CR(m)$ stands for Cost Ratio and is only function of the fidelity level, because is the computational cost ratio between high- and low-fidelity model. It assumes the same expression previously seen for MFEI.
The term $\eta(\boldsymbol{x}, m)$ is the sample density function, whose assessment has been drawn from [86]. It describes the density of datapoints from the fidelity level m and helps to prevent their clustering. Its formulation derives from the computation of the spatial distance between the sequential point and observed samples, made by using

the Gaussian correlation model. Therefore, its contribution within the formulation, drops rapidly when the sample x is close to a previously evaluated point. This happens because, the correlation function $R\left(\boldsymbol{x}, \boldsymbol{x}_i^{(m)}\right)$ converges from zero to one, as the spacial distance decreases. This is an essential requirement to avoid wasting resources on unnecessary samples.

4.1.3 Multi-Fidelity Predictive Entropy Search

According to [87], Multi-fidelity Predictive Entropy Search (MFPES) facilitates the non trivial trade-off between exploitation and exploration over the objective and the auxiliary functions. The index t represents the target function, whereas x_{*m} is the maximizer of the m-th function. The aim is to maximize information gain of only the global target maximizer x_{*t}, on the basis of next input observation $\langle x, m\rangle$.

$$\text{MFPES}(y_x, \langle x, m\rangle) = H(y_m(x)|y_X) - E_{p(x*_t|y_X)}[H(y_m(x)|y_X, x_{*t})]$$

where

$$H(y_m(x)|y_X) \triangleq 0.5\, log(2\pi e(\sigma^2_{\langle x,m\rangle\,|X} + \sigma^2_{nm}))$$

$$E_{p(x*_t|y_X)}[H(y_m(x)|y_X, x_{*t})] \approx -\frac{1}{2S}\sum_{s=1}^{S} log\left(v^{(s)}_{f_m} + \sigma^2_{nm}\right)$$

with $\sigma^2_{\langle x,m\rangle\,|X} \triangleq \sum_{\{\langle x,m\rangle\}\{\langle x,m\rangle\}|X}$ .The vector y_x collects the observed noisy outputs
The first term $H(y_m(x)|y_X)$ is the Gaussian predictive/posterior term and can be computed analytically.
The second term represents the predictive entropy conditioned on target maximizer and is calculable only through approximations that are widely described in [87].

4.1.4 Multi-Fidelity Max-value Entropy Search

Multi-fidelity Max-value Entropy Search (MFMES) is an acquisition function based on the information-gain criterion: it aims to maximize the mutual information between the queried points and the location of the global optimum of the highest fidelity function, in order to select the next iterate. MFMES acquisition function adopted in this thesis, refers to that proposed in 2020 [83], which is inspired by the single fidelity formulation made by Wang and Jegelka in 2017 [81], explained in the previous chapter. This method aims to facilitate the computation, making reliable the evaluation of the entropy. MES measures the information gain about the maximum value $f_* \coloneqq max_x f(\boldsymbol{x})$ which is in one-dimensional space: this facilitates the estimation of the mutual information and reduces computational costs. This is the main advantage of the method with respect to other information-gain and entropy search-based methods, such as Entropy Search and Predictive Entropy Search. They are based on the information maximization about the optimal solution $\boldsymbol{x}_* \coloneqq argmax_x f(\boldsymbol{x})$ of the unknown function, that is a multidimensional domain. They can identify a good query point with few iterations, but this involves expensive computations. For this reason, in high dimensions, where a large number of initial samples are required, they become quickly inefficient. Furthermore they characterize the uncertainty about $\boldsymbol{x}_*$ with a formulation analytically intractable, which has

to be approximated via expensive computations. Moreover, the optimum may not be unique, adding further complexity to this d-dimensional distribution [83]. Therefore, MES merges the ease of ES and PES of finding the next query points, with the reduced computational cost of its innovative formulation. Its performance is showed in terms of total cost of functions evaluations, but future works are necessary to evaluate performances based on wall-clock time. Furthermore, unlike classical evaluation measures such as expected improvement, it promises a measure of global utility which does not require any additional exploit-explore trade-off parameter [81]. Since objective functions have a variety of sampling costs, in a MFBO context, queries occur asynchronously. While developers of the method propose both Sequential and Parallel Querying approaches, in this thesis only the first one is implemented. It considers the case that a query is sequentially issued after the previous one is observed. We assume the formulation expressed at the beginning of the chapter. Given a training data set D_t, the acquisition function is defined as:

$$\text{MFMES}(\boldsymbol{x}, m) \coloneqq I(f_*; f_x^{(m)}|D_t)/\lambda^{(m)}$$

where $I(f_*; f_x^{(m)}|D_t)$ is the mutual information between f_* and $f_x^{(m)}$. Maximizing $\text{MFMES}(\boldsymbol{x}, m)$, allows to obtain a pair of the input $\boldsymbol{x}$ and the fidelity m, which maximally gains information about the optimal value f_* of the highest fidelity function per unit cost. The mutual information can be written as the difference of the entropy:

$$I(f_*; f_x^{(m)}|D_t) = H(f_x^{(m)}|D_t) - E_{f_*|D_t}[H(f_x^{(m)}|f_*, D_t)] \tag{4.8}$$

where $H(\cdot|\cdot)$ is the conditional entropy of $p(\cdot|\cdot)$. The first term on the right, can be derived analytically for any fidelity m :

$$H(f_x^{(m)}|D_t) = \log\left(\sigma_x^{(m)}\sqrt{2\pi e}\right) \tag{4.9}$$

where $e \coloneqq exp(1)$. The second term takes the expectation over the maximum f_*. An analytical formula for this expectation is not available, so it is approximated using Monte Carlo estimation by selecting F_* , that is a set of function maxima f_* sampled from the GPR:

$$E_{f_*|D_t}[H(f_x^{(m)}|f_*, D_t)] \approx \sum_{f_* \in F_*} \frac{H(f_x^{(m)}|f_*, D_t)}{|F_*|}$$

Actually, when this function has been implemented in the optimization algorithm adopted for this thesis, that solves a minimization problem, and not a maximization one, F_* is a set of function minima.
Monte Carlo simulations are here adopted because they can sample from probability distributions to study phenomena that do not have direct analytical solutions.
As said before, two strategies are proposed by Wang & Jegelka to sample the max value: sampling from a Gumbel distribution or sampling the highest fidelity function from a posterior Gaussian distribution (Random Feature Map, RFM). Here, the second choice is preferred due to the GPR model implemented in previous works. For a sampled f_*, it is necessary to calculate the entropy of $p(f_x^{(m)}|f_*, D_t)$ in equation (4.8) as $p(f_x^{(m)}|f_x^{(M)} \leq f_*, D_t)$, in order to condition only on the given $\boldsymbol{x}$ rather than requiring $f_x^{(M)} \leq f_*$ for $\forall \boldsymbol{x} \in \mathcal{X}$. This simplification makes the computation more tractable, leading to superior performances, and has been employed by most of the

other entropy-based BO methods [77, 81].
For any $\zeta \in \mathbb{R}$, define

$$\gamma_{\zeta}^{(m)}(\boldsymbol{x}) \coloneqq (\zeta - \mu_x^{(m)})/\sigma_x^{(m)} \tag{4.10}$$

as a function for scaling.
When $m = M$, the density function $p(f_x^{(m)}|f_x^{(M)} \leq f_*, D_t)$ is a truncated normal distribution, whose entropy can be represented as [88]:

$$H(f_x^{(M)}|f_x^{(M)} \leq f_*, D_t) = \log\left(\sqrt{2\pi e}\sigma_x^{(M)}\Phi(\gamma_{f_*}^{(M)}(\boldsymbol{x}))\right) - \frac{\gamma_{f_*}^{(M)}(\boldsymbol{x})\phi(\gamma_{f_*}^{(M)}(\boldsymbol{x}))}{2\Phi(\gamma_{f_*}^{(M)}(\boldsymbol{x}))} \tag{4.11}$$

where ϕ and Φ are the probability density function and the cumulative distribution of the standard normal distribution.
For the information gain from a lower fidelity, when $m \neq M$, the entropy cannot be calculated by representing a conditional distribution of $f_x^{(m)}$ given f_* as a truncated normal distribution. The joint conditional distribution $p(f_x^{(M)}|f_x^{(m)}, D_t)$, which is constructed from the two dimensional GPR predictive distribution $p(f_x^{(M)}, f_x^{(m)}|\boldsymbol{x}, D_t)$, can be obtained as:

$$f_x^{(M)}|f_x^{(m)}, D_t \sim \mathcal{N}(u(\boldsymbol{x}), s^2(\boldsymbol{x}))$$

where

$$\begin{aligned} u(\boldsymbol{x}) &= \sigma_x^{2(mM)}(f_x^{(m)} - \mu_x^{(m)})/\sigma_x^{2(m)} + \mu_x^{(M)} \\ s^2(\boldsymbol{x}) &= \sigma_x^{2(M)} - (\sigma_x^{2(mM)})^2/\sigma_x^{2(m)} \end{aligned} \tag{4.12}$$

so the entropy of $p(f_x^{(m)}|f_x^{(M)} \leq f_*, D_t)$ can be written as:

$$H(f_x^{(m)}|f_x^{(M)} \leq f_*, D_t) = -\int Z\Psi(f_x^{(m)}) \log\left(Z\Psi(f_x^{(m)})\right) df_x^{(m)} \tag{4.13}$$

where

$$\begin{aligned} Z &\coloneqq 1/\sigma_x^{(m)}\Phi(\gamma_{f_*}^{(M)}(\boldsymbol{x})) \\ \Psi(f_x^{(m)}) &\coloneqq \Phi((f_* - u(\boldsymbol{x}))/s(\boldsymbol{x}))\phi(\gamma_{f_x^{(m)}}^{(m)}(\boldsymbol{x})). \end{aligned} \tag{4.14}$$

In this way, the entropy is represented through a one dimensional integral over $f_x^{(m)}$ and standard numerical integration techniques can be used to approximate it with simple computations. For a given f_*, the integral can be computed as $O(1)$.

4.2 Numerical implementation

This section describes the implementation in Matlab of the multifidelity optimization algorithm, which is then used for all the test cases, subjects of this thesis. The optimization solves a minimization problem. Following the flowchart in Figure 4.1, it is possible to summarize the main steps of the algorithm. The first step provides for setting up the structure *opt*, that represents the optimization setup,

and the structure *fun*, that collects objective function modeled with different increasing fidelities $m = 1, \ldots, M$. The structure *Opt* includes: the dimension of the design space $\mathcal{X}$, the range in which the design variables change, the maximum number of iterations, the maximum budget geared for all the optimization process,

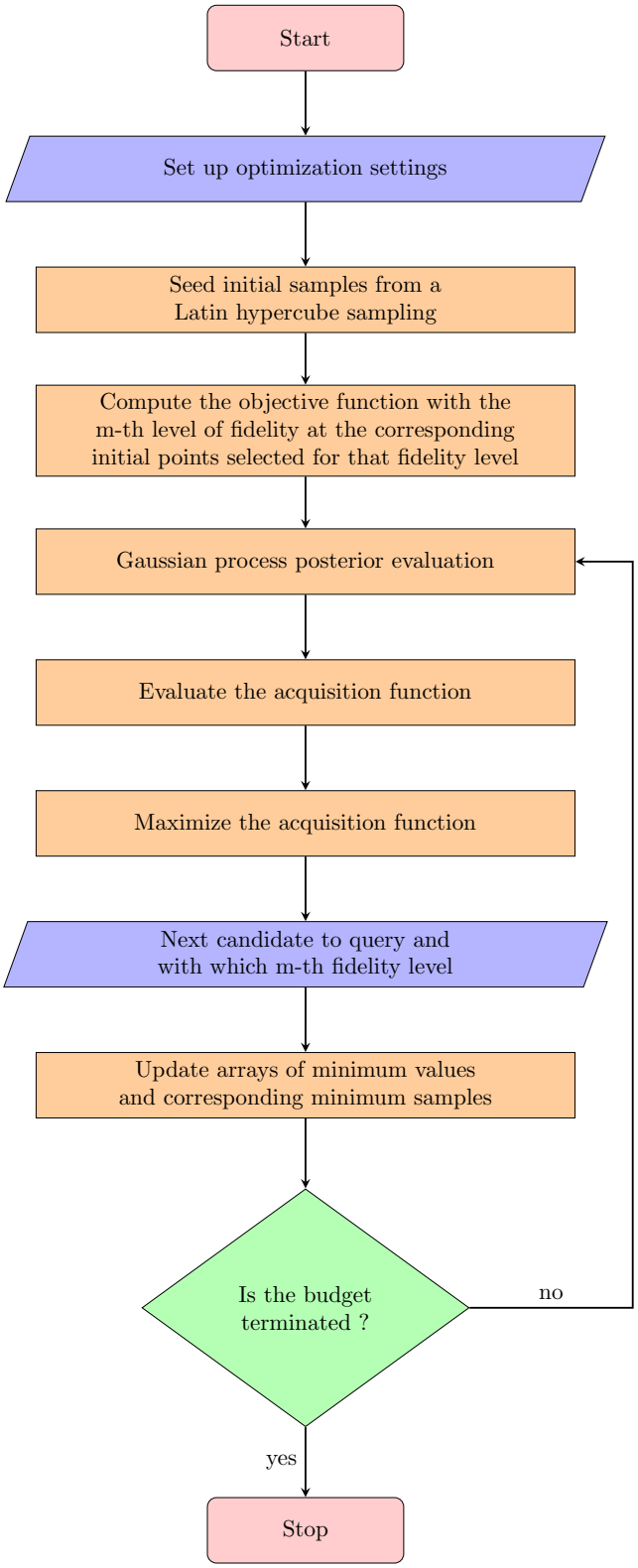


Figure 4.1: MFBO algorithm implemented

the size of the grid which discretizes the space of design, the number of different fidelities used and the corresponding cost $\lambda^{(m)}$ for each call (generally, the cost to evaluate the high-fidelity objective function is one, whereas lower-fidelities costs are set proportionally), the number of initial design points and how they are divided among the different fidelity levels, the amount of Monte Carlo simulations to use (it is necessary only if the selected acquisition function is MFMES), the number of tests to run for statistical purposes and the type of multifidelity acquisition function (MFEI,MFMES or MFPI). Then, a Latin hypercube sampling (lhs) method is adopted for spreading the sample points into the design domain and an initial grid of candidates is drawn from it randomly. After selecting the initial points, the m-th functions are computed and their outputs are collected in the array $y^{(m)}$. Successively, the minimum value, *minvalues*, and its corresponding sample point x, *minsamples*, of the high-fidelity function are evaluated. Before starting the iterative process of optimization, the budget that concerns the initial points evaluation, is computed. Then, a *for* cycle is set to start the optimization until the end of budget is met or, eventually, the maximum number of iterations is achieved (generally, the budget limitation is selected to be more restrictive than the limitation on the iterations). Within the *for* loop, the surrogate model is built, on the basis of the first iterations, and progressively updated. The surrogate is set up by fitting a Multifidelity Gaussian Process regression model (MFGP), in order to have a prediction for unsampled points. It is trained using samples and response values, computed at previous iterations, made on all the m-th fidelities. It adopts a squared exponential kernel function. Therefore, the posterior is obtained in terms of mean function $\mu^{(m)}$ and variance $\sigma^{2(m)}$ from the Gaussian process. Successively, the acquisition function is computed and maximized: the next point to query $x_{i+1}^{(m)}$ and with which fidelity function, are so determined and, then deleted from the initial design space, avoiding replicates. A description of the algorithms implemented for each acquisition function, can be found in tables (4.3, 4.1, 4.2). Then, *minvalues* and *minsamples* arrays are updated and budget too. The cycle restarts.
For all the experiments runs that will be shown in the next chapters (Chapters 5 and 6), computational resources were provided by HPC@POLITO (http://hpc.polito.it).

Algorithm: MFEI

1. **for** i=1: maximum number of iterations (or end of budget)
2. **function**$[x_{i+1}, M, \mathcal{X}]$= MFEI$(\mu_x^{(m)}, \sigma_x^{(m)}, f^{(M)}, \mathcal{X}, \lambda^{(m)})$
3. Compute $f^{(M)}(\boldsymbol{x}^+) = min(f^{(M)}(\boldsymbol{x}))$
4. **for** $m = 1 : M - th$ fidelity level
5. Compute $\alpha_1(\boldsymbol{x}, m) = \text{corr}\left[f^{(m)}(\boldsymbol{x}), f^{(M)}(\boldsymbol{x})\right]$
6. Compute $\alpha_2(\boldsymbol{x}, m) = 1 - \frac{\sigma_\epsilon}{\sqrt{\sigma^{2(m)}(\boldsymbol{x}) + \sigma_\epsilon^2}}$
7. Compute $\alpha_3(m) = \frac{\lambda_M}{\lambda_m}$
8. **end for**
9. Calculate EI= $E\left[max\left(f^{(M)}(\boldsymbol{x}^+) - f^{(M)}(\boldsymbol{x}), 0\right)\right]$ (Eq.4.2)
10. Calculate MFEI$(\boldsymbol{x}, m) = EI \cdot \alpha_1(\boldsymbol{x}, m) \cdot \alpha_2(\boldsymbol{x}, m) \cdot \alpha_3(m)$ (Eq.4.1)
11. $[x_{i+1}, M] = argmax$(MFEI)
12. Remove the new point selected, from the space of design $\mathcal{X}$
13. **end function**
14. Update the surrogate by evaluating the M-th model at x_{i+1}
15. Evaluate the current minimum of the objective function (*minsamples, minvalues*)
16. **end for** on i

Table 4.1: MFEI algorithm

Algorithm: MFPI

1. **for** i=1: maximum number of iterations (or end of budget)
2. **function**$[x_{i+1}, M, \mathcal{X}]$= MFPI$(\mu_x^{(m)}, \sigma_x^{(m)}, f^{(M)}, \mathcal{X}, \lambda^{(m)})$
3. Compute $f^{(M)}(\boldsymbol{x}^+) = min(f^{(M)}(\boldsymbol{x}))$
4. **for** $m = 1 : M - th$ fidelity level
5. Compute $Corr(\boldsymbol{x}, m) = Corr\left[f^{(M)}(\boldsymbol{x}), f^{(m)}(\boldsymbol{x})\right] \Phi\left(\frac{f^{(M)}(\boldsymbol{x}^+) - f^{(M)}(\boldsymbol{x})}{\sigma^{(M)}(\boldsymbol{x})}\right)$ (Eq.4.5)
6. Compute CR(m) $= \frac{\lambda_M}{\lambda_m}$ (Eq.4.6)
7. **end for**
8. Compute $\eta(\boldsymbol{x}, m) = \prod_{i=1}^{n}\left[1 - R\left(\boldsymbol{x}, \boldsymbol{x}_i^{(m)}\right)\right]$ (Eq.4.7)
9. Calculate $PI(\boldsymbol{x}) = \Phi\left(\frac{f^{(M)}(\boldsymbol{x}^+) - f^{(M)}(\boldsymbol{x})}{\sigma^{(M)}(\boldsymbol{x})}\right)$ (Eq.4.4)
10. Calculate MFPI$(\boldsymbol{x}, m) = PI(\boldsymbol{x}) \times Corr(\boldsymbol{x}, m) \times CR(m) \times \eta(\boldsymbol{x}, m)$ (Eq.4.3)
11. $[x_{i+1}, M] = argmax$(MFPI)
12. Remove the new point selected, from the space of design $\mathcal{X}$
13. **end function**
14. Update the surrogate by evaluating the M-th model at x_{i+1}
15. Evaluate the current minimum of the objective function (*minsamples, minvalues*)
16. **end for** on i

Table 4.2: MFPI algorithm

Algorithm: MFMES

1. **for** i=1: maximum number of iterations (or end of budget)
2. **function**$[x_{i+1}, M, \mathcal{X}]$=MFMES$(\mu_x^{(m)}, \sigma_x^{(m)}, f_x^{(m)}, \mathcal{X}, \lambda^{(m)})$
3. Calculate $H_0 = \log(\sigma_x^{(m)}\sqrt{2\pi e})$ (Eq.4.9)
4. **for** $m = 1 : M - th$ fidelity level
5. **for** $j = 1$: number of Monte Carlo simulations
6. Compute $F^* = min(f_*^{(M)})$
7. Compute $\gamma_{f_*}^{(M)}(\boldsymbol{x})$, $\phi(\gamma_{f_*}^{(M)}(\boldsymbol{x}))$, $\Phi(\gamma_{f_*}^{(M)}(\boldsymbol{x}))$ (Eq.4.10)
8. **if** m=M
9. Calculate $H_1 = \sum_{f_* \in F_*} \frac{H(f_x^{(M)} | f_*^{(M)} \leq f_*, D_t)}{|F_*|}$ (Eq.4.11)
10. **else**
11. Compute $Z, \sigma_x^{2mM}, u(\boldsymbol{x}), s^2(\boldsymbol{x}), \gamma_{f_x^{(m)}}^{(m)}, \Psi(f_x^{(m)})$ (Eq.4.12,4.14)
12. Calculate $H_1 = \sum_{f_* \in F_*} \frac{H(f_x^{(m)} | f_*^{(M)} \leq f_*, D_t)}{|F_*|}$ (Eq.4.13)
13. **end if**
14. **end for** on j
15. **end for** on m
16. MFMES= $(H_0 - H_1)/\lambda^{(m)}$
17. $[x_{i+1}, M] = argmax$(MFMES)
18. Remove the new point selected, from the space of design $\mathcal{X}$
19. **end function**
20. Update the surrogate by evaluating the M-th model at x_{i+1}
21. Evaluate the current minimum of the objective function (*minsamples, minvalues*)
22. **end for** on i

Table 4.3: MFMES algorithm

Chapter 5

Applications to L1 problems

In this chapter L1 problems, proposed as standards by dedicated working groups of the international community, are used for the assessment of the previously described multifidelity methods. The term L1 represents the simplest category of problems, that is analytic functions, which have exact evaluations, negligible computational costs and a little engineering utility [89], because they exemplify mathematical properties and behaviours of real engineering applications. Nevertheless, they are useful to stress and characterize the methods employed in terms of strengths and weaknesses, and to demonstrate proof of concept.
L1 benchmarks are selected with the aim to capture fundamental mathematical aspects and behaviours. The selected set of functions comprises: the Sinusoidal-squared function, the Forrester function, the Rosenbrock function, the Rastrigin function shifted and rotated, the Clark and Bae function and the coupled Spring-Mass System. They allow to test the multifidelity methods in context that include multimodal problems with local optima and increasing dimensionality of the objective function: key features of design and optimization engineering systems.
For each function, it will be a mathematical description of the models and then an analysis of the results obtained from their implementations.
Two fidelity levels will be adopted, the best and the worst model presented in the guidelines proposed by NATO STO, where $m = 1$ stands for the highest fidelity, and $m = 2$ for the lowest one. A table will summarise the initial settings used for the experiments, such as the initial budget, the evaluation costs of each function and the number of initial samples for each fidelity model.
A budget is used as a termination criterion for simulating resource constraints that affect real-world design problems. It is set proportionally to the dimensionality D, as: $Budget = 100D$; the fidelity cost assignment is determined considering $\lambda^1 = 1$ for all the high-fidelity functions. Therefore, the cost assigned to the low-fidelity function is a fraction of the computational cost of $f_1(\boldsymbol{x})$. The amount of Monte Carlo simulations, required by the Multifidelity Max-value Entropy Search, is set up to 10 for all the benchmark problems, following the developers' settings [83].
The optimization performances of the methods are, then, evaluated in terms of function relative error with respect to the objective function (i.e. normalized difference in the function space) Δ_f. It is computed as: $\Delta_f = \frac{f(\boldsymbol{g})-f^*_{min}}{f^*_{max}-f^*_{min}}$ where $f(\boldsymbol{g})$ is the minimum found by the algorithm, f^*_{min} is the analytical minimum and f^*_{max} is the analytical maximum of the function $f(\boldsymbol{x})$ in the domain of design. Then, the median of Δ_t and the median of the total costs are evaluated. The values are computed

taking into account ten tests, for each method and for each benchmark problem for statistical purposes (except for the Rosenbrock 5D function and the spring-mass system, where the tests are reduced to just five). The global performance of the methods Δ_t are assessed to take into account not only the error about the function, but also that about the location of the optimum found:

$$\Delta_t = \sqrt{\frac{\Delta_x^2 + \Delta_f^2}{2}}$$

$$\Delta_x = \sqrt{\frac{1}{N}\sum_{j=1}^{N}\left(\frac{g_j - x^*_{j,min}}{Z_j}\right)^2}$$

represents the normalized Euclidean distance between the location of the global optimum found with the optimization process (g) and the analytical minimizer x^*_{min}, with $Z_j = |u_j - l_j|$ is the range within the j-th variable changes.
Furthermore, also the Confidence Interval (CI) is computed. It represents a limited interval in which it is possible to find the true value, with a certain likelihood (confidence grade). The bounds are set to 25 % and 75% percentile of the Gaussian distribution over the median so all the observations that are part of this interval are accepted, while the others are excluded in order to avoid outliers.

5.1 Forrester function

The Forrester function [90] is a 1D non linear function over the domain [0,1]. The following equations describe the high- and low-fidelity functions:

$$f_1(\boldsymbol{x}) = (6x - 2)^2 \sin(12x - 4)$$

$$f_2(\boldsymbol{x}) = 0.5 f_1(\boldsymbol{x}) + 10(x - 0.5) - 5.$$

Its minimum is $f(\boldsymbol{x^*}) = -6.0207$ and is located at $\boldsymbol{x^*} = 0.7572$. The function stresses the research for the minimum, thanks to its multimodality. It helps to check if the methods tend to stop in a local minimum.

Function	Dimension	Budget	HF cost	LF cost	HF samples	LF samples
Forrester	1	100	1	0.05	2	5

Table 5.1: Forrester experiment setup

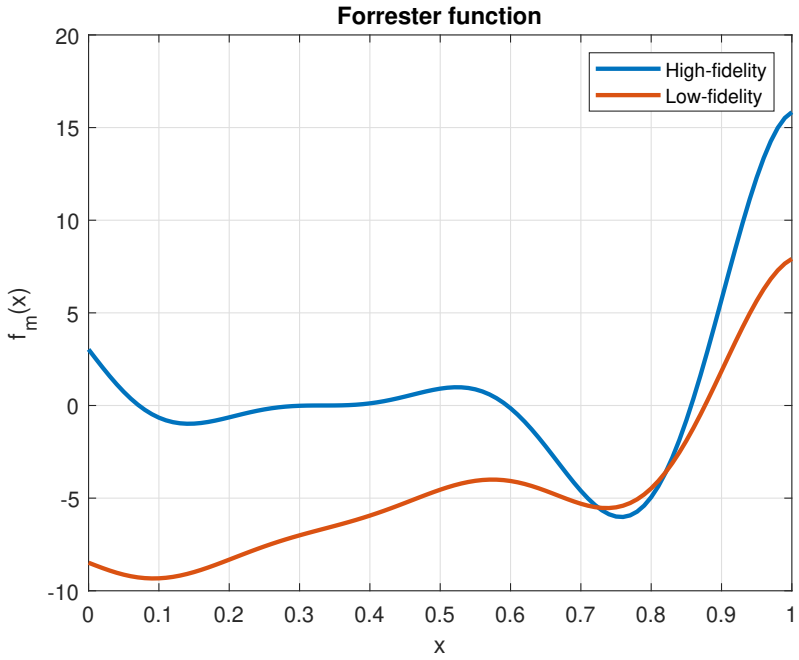


Figure 5.1: Forrester function

5.1.1 Results

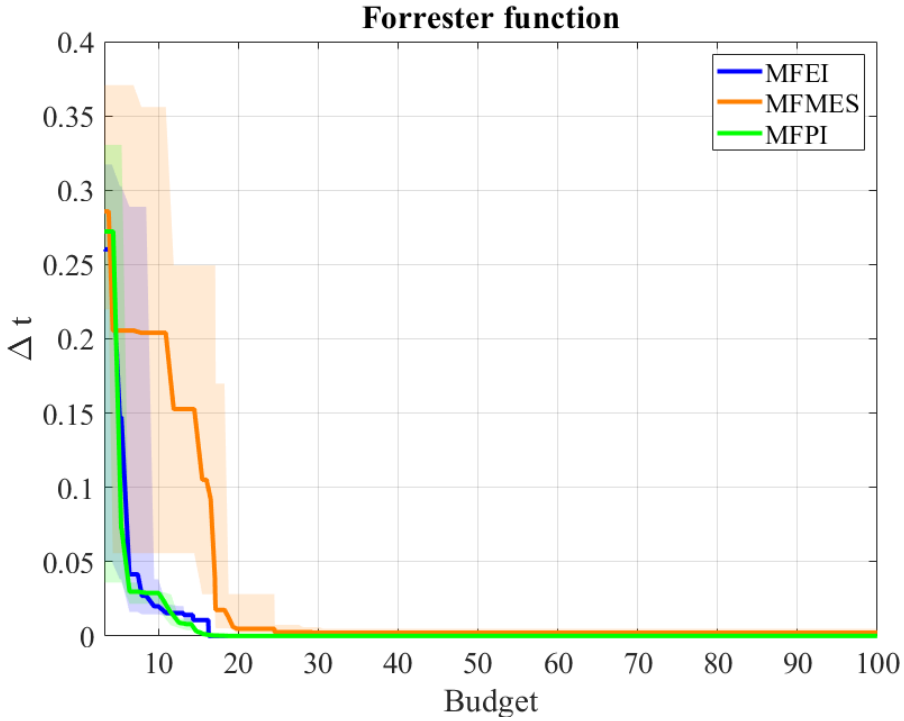
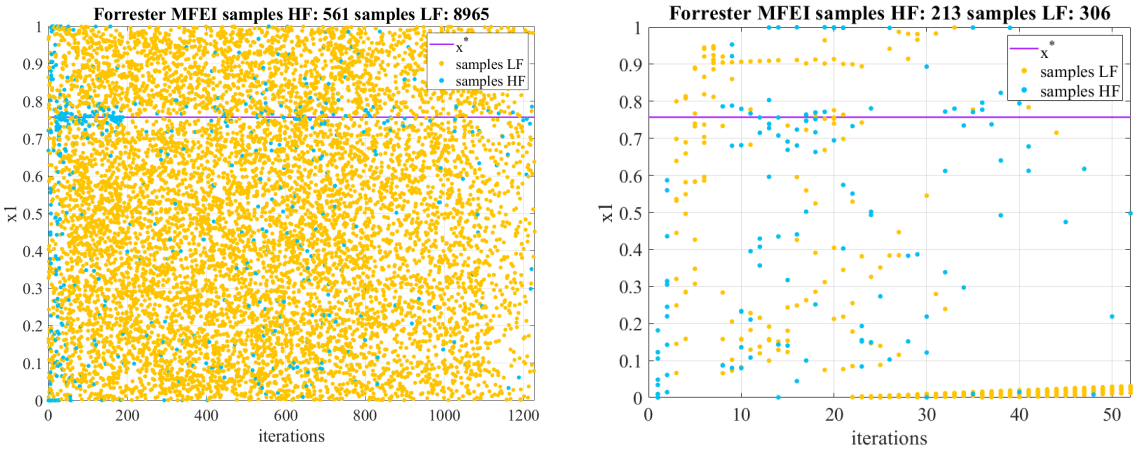


Figure 5.2: Comparison of MFEI, MFMES and MFPI on Forrester function.

The figure above is a representation of the median values, along with all the other object observations falling into the confidence interval (shaded areas), evaluated by the three different methods. It is possible to observe that MFPI, initially decreases the error more rapidly than the other methods but, at around budget=8, its perfor-

mance reduces and swings between a better and a worst behaviour than the MFEI. However, on the population of tests executed, the minimization of the error takes place with a budget of around 16.3 for the MFEI, and 23 for the MFPI. The error can be reduced quickly, thanks also to the great accuracy of the surrogate, which, with low dimensionality, can achieve high performances, even with a narrow number of initial samples. For what concerns MFMES, the error cannot be minimized beyond the value of 0.002, that is achieved with a budget of approximately 47.4.

The minimum $x^* = 0.7572$ is approximately reached after 52 iterations of MFEI,



(a) Complete

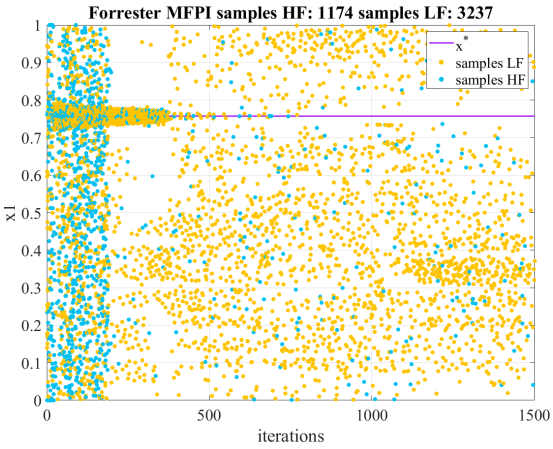
(b) Until the optimum

Figure 5.3: Distribution of MFEI samples over x till the optimum.

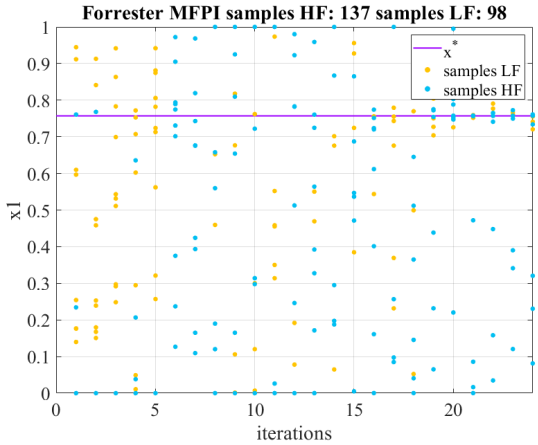
with the use 213 and 306 patterns of high- and low-fidelity, respectively. The quite low number of iterations required is due to the great surrogate accuracy on a 1D problem. This allows to find rapidly the global optimum. Later, the behaviour of the algorithm is that to analyse the design domain with both exploration and exploitation purposes, but preferring exploration: a quite uniform distribution of low-fidelity samples is spread over the domain. Therefore, the number of LF samples (8965) is much greater than that of high-fidelity (561).

Multifidelity Probability of Improvement, finds the global optimum after around 24 iterations, by means of 137 HF calls and 98 LFs. This great performance is still due to the good approximation of the objective function. Then a cone of points clusters around x^*, because this acquisition function tends to prefer exploitation, therefore the algorithm continues to seed points in an already good region, in order to continuously try to find a better point. The total number of high-fidelity samples used, is substantially greater than that of MFEI (around twice as much), with a greater concentration at the first 200 iterations, always with the aim to give more importance to the exploitation towards the optimum, than to the exploration to refine an already accurate surrogate.

Multifidelity Max-value Entropy Search, reaches the optimum location almost around the 104-th run, by calling quite an equal number of high- and low-fidelity functions. In particular the method employs 506 and 505 samples of HF and LF. After finding the minimum, the space of design is discovered by sampling also HF samples, but mainly LF ones, along some trajectories. The preference of the LF patterns, is due to the great amount of budget consumed in the first 230 iterations on high-fidelity.

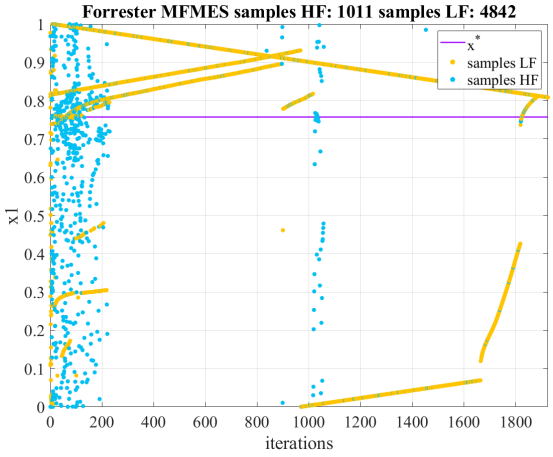


(a) Complete

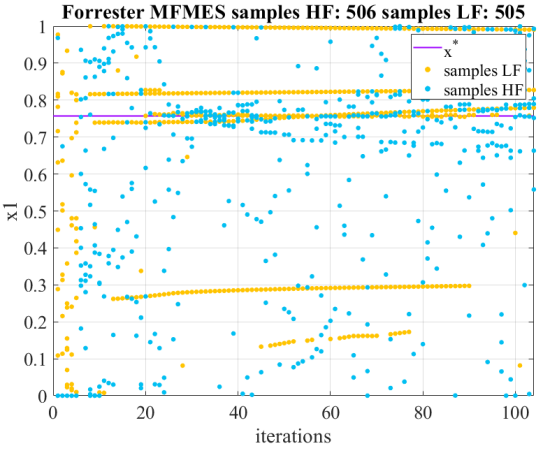


(b) Until the optimum

Figure 5.4: Distribution of MFPI samples over x.



(a) Complete



(b) Until the optimum

Figure 5.5: Distribution of MFMES samples over x.

5.2 Sinusoidal squared 1D function

The function [91] domain lies in the interval [0,1]. The high- and low-fidelity functions are defined respectively as:

$$f_1(\boldsymbol{x}) = (x - \sqrt{2})(f_2(\boldsymbol{x}))^2$$

$$f_2(\boldsymbol{x}) = \sin(8\pi\boldsymbol{x}).$$

Therefore, the high-fidelity objective function is a non linear function of the low-fidelity counterpart. The minimum is $f(\boldsymbol{x}^*) \simeq -1.35201$ at $\boldsymbol{x}^* \simeq 0.0619147$. This problem has been set as a benchmark, because it allows to evaluate the responses of the methods when subjected to a 1D multimodal problem.

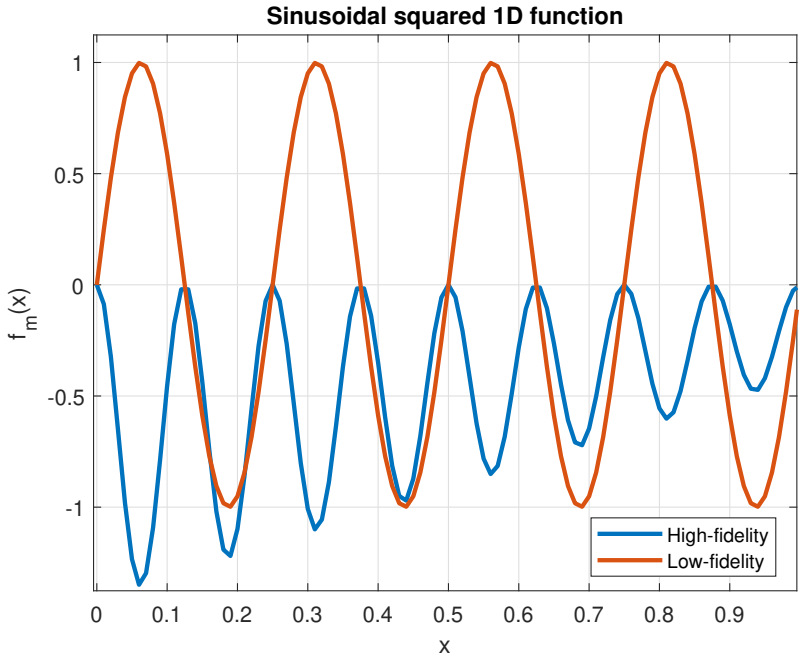


Figure 5.6: Sinusoidal squared 1D function

Function	Dimension	Budget	HF cost	LF cost	HF samples	LF samples
Sinusoidal sqrd	1	100	1	0.05	2	5

Table 5.2: Sinusoidal squared 1D experiment setup

5.2.1 Results

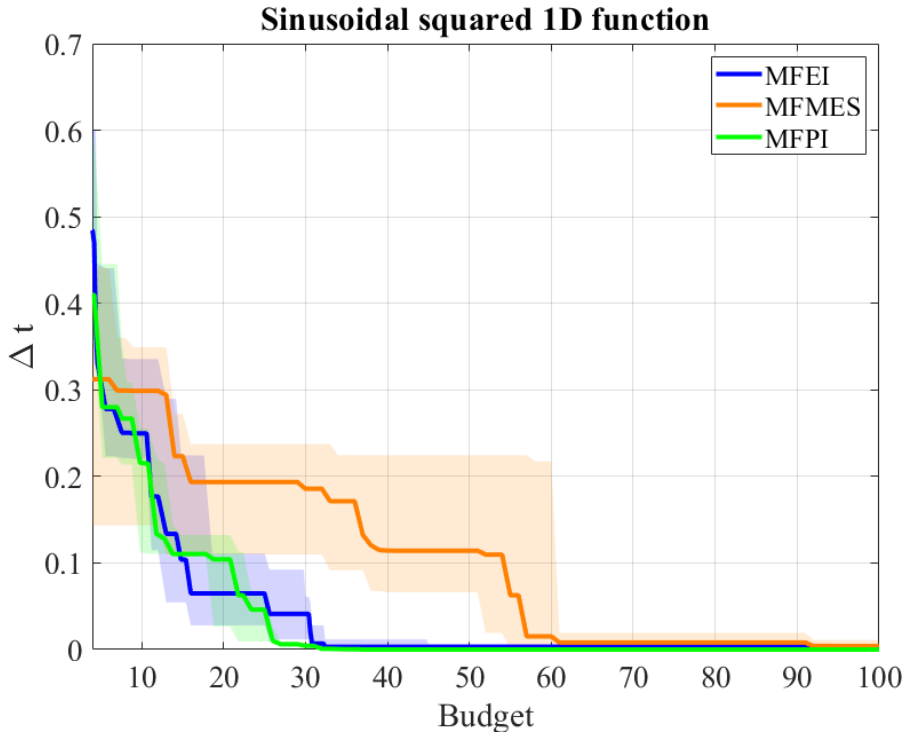
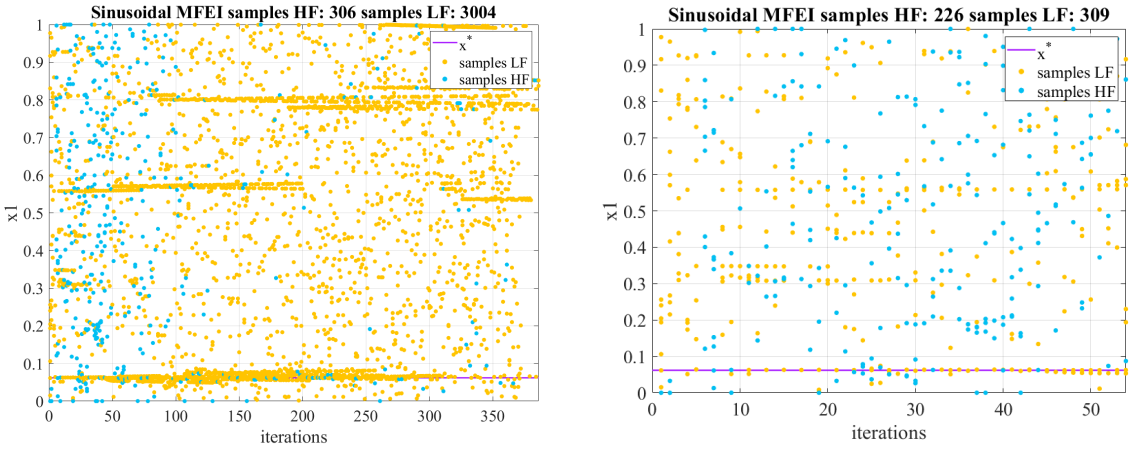


Figure 5.7: Comparison of MFEI, MFMES and MFPI on Sinusoidal squared 1D function.

All the methods present a typical error reduction by steps. It is due to the main characteristic of this problem. It employs a low-fidelity function that starts nearly in opposition of phase with the high fidelity, thus stressing the method to check if it rests trapped in a local minimum. MFPI results the method with the best behaviour, indeed, it allows to reduce the function error with a budget of 37, while MFEI employs a budget of 32, but stopping with a residual error of 0.003. Whereas MFMES cannot reduce the error beyond 0.004 achieved with budget 92.
Multifidelity Expected Improvement initially chooses to use the low-fidelity function above all. Then, between the 8-th run and the 50-th, it places a higher amount of

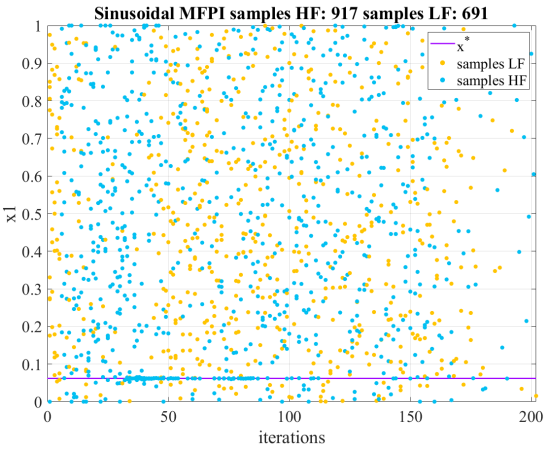


(a) Complete

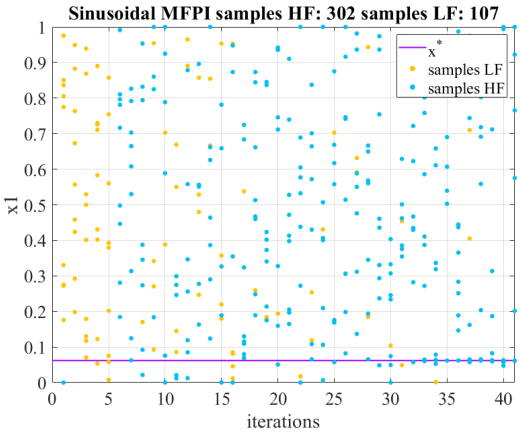
(b) Until the optimum

Figure 5.8: Distribution of MFEI samples over x.

HF calls. A slight thickening of HF points can be seen around x=0.2, that is the location of the greater local minimum. Then, the search space is fulfilled of LF calls, and trajectories of mainly low-fidelity samples can be found around the other local minima of the objective function (at x=0.56 and x=0.8), with a greater extension towards the target location x^*.
MFPI starts selecting a higher number of LF samples, then a concentration of HF patterns can be found until the location of the optimum is reached. The Probability of Improvement, employs 41 runs to achieve the optimum, by seeding 302 high-fidelity samples and 107 low-fidelity. Then, the samples cluster around the x^* and both high and low-fidelity functions are widely called. As said, this method tends to adopt a higher amount of high-fidelity samples, in order to promote exploitation. Remembering its analytical formulation, the term $Corr(\boldsymbol{x}, m)$ introduces a penalty when calling the low-fidelity function, that is a removal from the aim of maximizing the acquisition function for finding the optimum location.
The MFMES first calls the low-fidelity function, then the algorithm focuses on the selection of nearly only high-fidelity. It seems that the method tends to spread points following search paths, especially for drawing the objective function local minima at $x = 0.8, x = 0.7$ and towards $x = 0.6$: there is a thickening of samples from both the right and the left limits of these minima. The convergence towards the global minimum is found with 92 iterations and with around 10 times more HF samples than the LF ones. After reaching the optimum, a dense amount of points is present towards the greater local minimum in x=0.2. Then, after around 110 iterations, the acquisition function places quite constantly the points, even by using



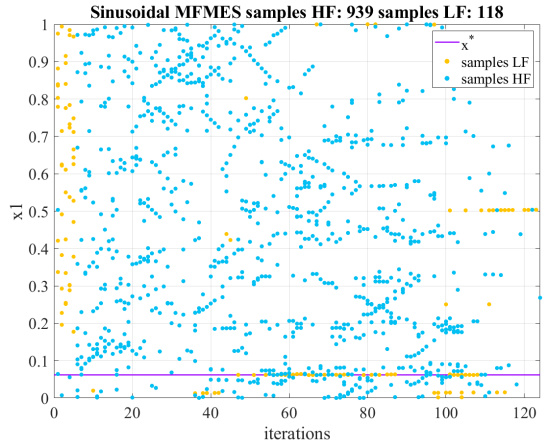
(a) Complete



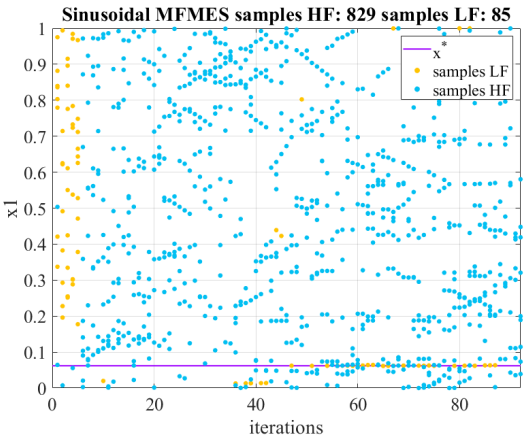
(b) Until the optimum

Figure 5.9: Distribution of MFPI samples over x.

the low-fidelity function. The Max-value entropy Search, is the method that tries to reduce, as much as possible, the total number of calls (1057 calls with respect to the 3310 and the 1608, respectively of the MFEI and MFPI). Indeed it spends more budget by using the HF function.



(a) Complete



(b) Until the optimum

Figure 5.10: Distribution of MFMES samples over x.

5.3 Rosenbrock 2D function

It is D-dimensional [92] non-convex function with domain in $[-2, 2]^D$, where D is the dimensionality chosen. It is defined as:

$$f_1(\boldsymbol{x}) = \sum_{i=1}^{D-1} 100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2$$

$$f_2(\boldsymbol{x}) = \frac{f_1(\boldsymbol{x}) - 4 - \sum_{i=1}^{D} 0.5x_i}{10 + \sum_{i=1}^{D} 0.25x_i}$$

The low-fidelity is obtained from a transformation of the high-fidelity function evaluations by linear additive and multiplicative factors [93]. It is a unimodal function whose global minimum is $f(\boldsymbol{x}^*) = 0$ at $\boldsymbol{x}^* = (1, ..., 1)$. Even if its global behaviour results easy to be approximated, the research of the minimum is not trivial, because it is located in a typical steep and narrow valley, with a slight gradient, in which the surrogate may be trapped [94].

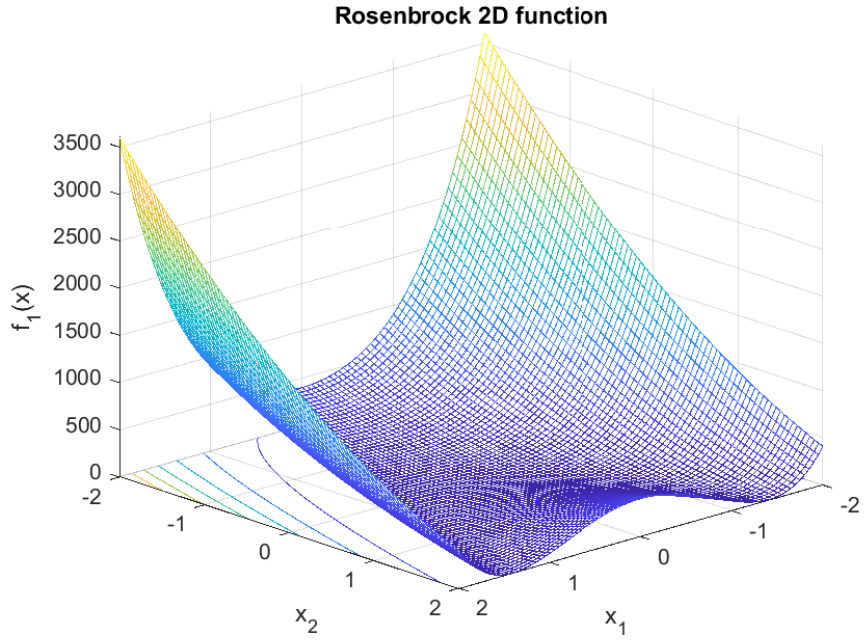


Figure 5.11: Rosenbrock 2D high-fidelity function

Function	Dimension	Budget	HF cost	LF cost	HF samples	LF samples
Rosenbrock	2	200	1	0.1	5	10

Table 5.3: Rosenbrock 2D experiment setup

5.3.1 Results

The Expected Improvement eliminates rapidly the error, spending a budget of 28.5. MFMES and MFPI both stop with a residual of one order of magnitude greater

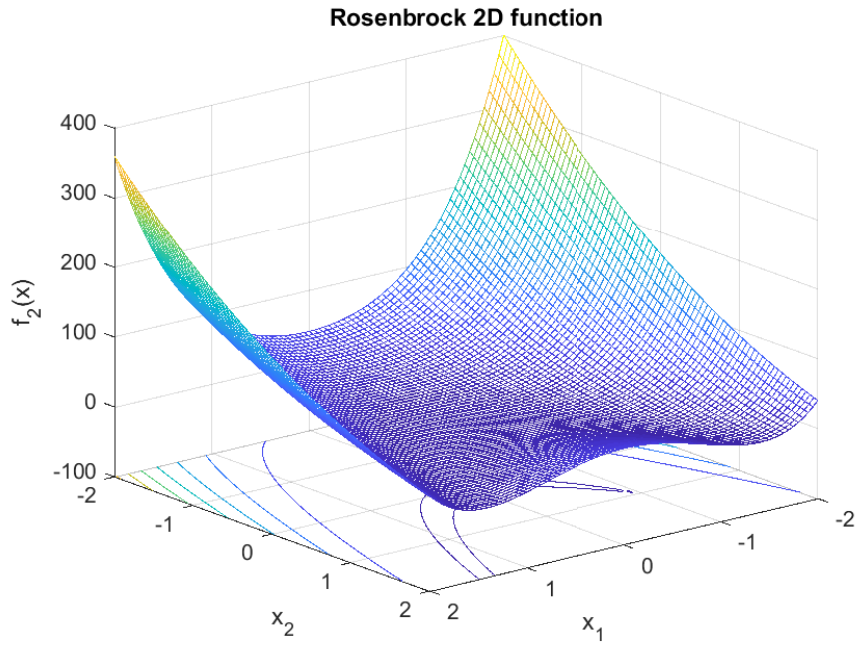


Figure 5.12: Rosenbrock 2D low-fidelity function

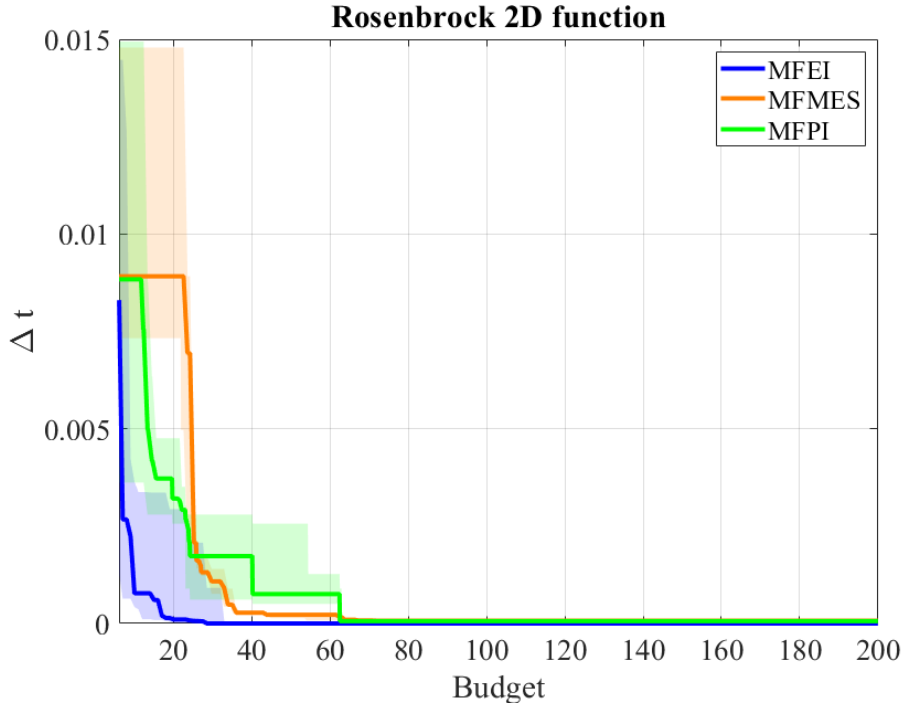
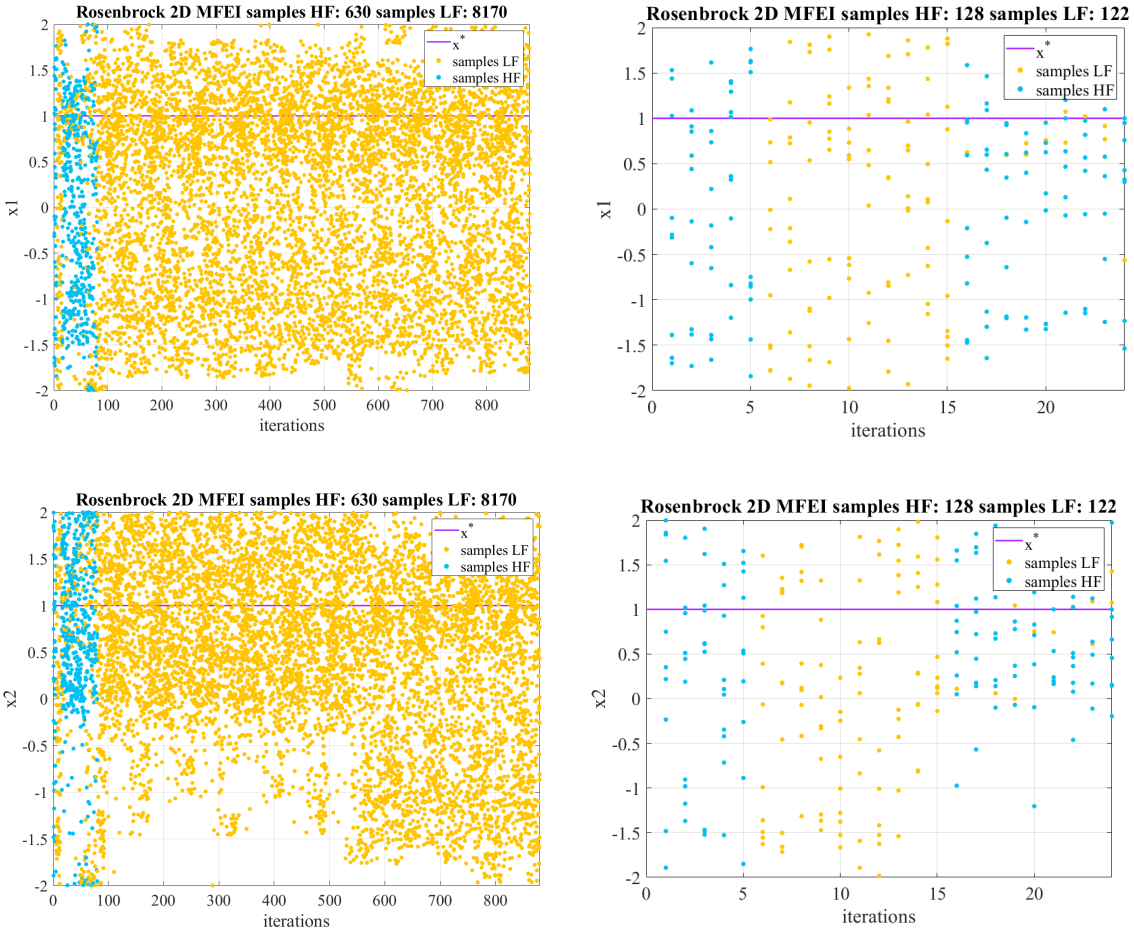


Figure 5.13: Comparison of MFEI, MFMES and MFPI on Rosenbrock 2D function.

than MFEI. It is achieved respectively, with a budget of 72 and 62. However, it is possible to observe that, even if MFPI starts with a better sampling, it then loses this advantage, so that the MFMES results more direct towards the optimum, starting from budget 26.
MFEI needs about 24 runs for finding x^*, thanks to the 128 HF patterns and the 122 ones of LF. Once the minimum is correctly located, the algorithm searches into the domain, first by calling the high-fidelity, because sufficient budget is still available;



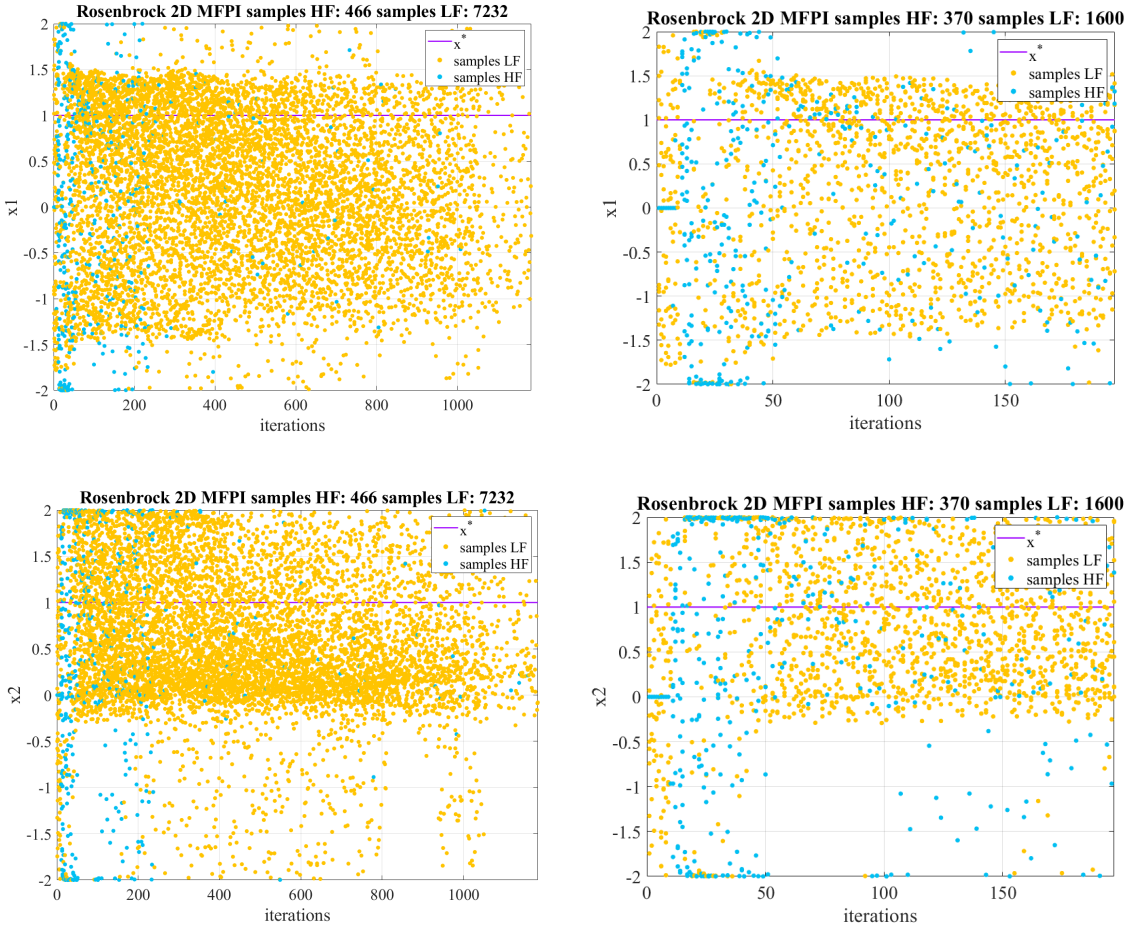
(a) Complete

(b) Until the optimum

Figure 5.14: Distribution of MFEI samples over x.

then, by using only LF: the number of LF patterns is about 13 times greater than that of HF. It is possible to notice that, from around the 100-th iteration, a series of sampling seems to keep happening every 100 iterations, thus searching within the valley of the minimum.
MFPI uses a mean of about 197 runs to find the two coordinates of the optimum. It especially employs LF samples: they amount to 1600 points, with respect to the 370 points of high-fidelity. The first iterations see a cluster of HF points towards the extremes of the search space, probably because of the initial lack of accuracy of the surrogate due to a disadvantageous initial sampling. Then, the method proceeds collecting samples above the zero coordinates, by calling essentially the low-fidelity model. Here the low-fidelity function assumes values very close to the minimum. The acquisition function spends the greatest part of the budget by using LF samples, indeed, the total number of low-fidelity patterns result about 15.5 times greater than the high-fidelity ones.
The less significant set of total patterns is adopted by the MES acquisition function, because it employs a greater number of HF patterns than the other two methods. It is possible to observe a net change of strategy of sampling: for the first ten runs, it implements the low-fidelity model, then it starts giving greater importance to the high-fidelity model allowing to achieve the optimum. After the convergence, it

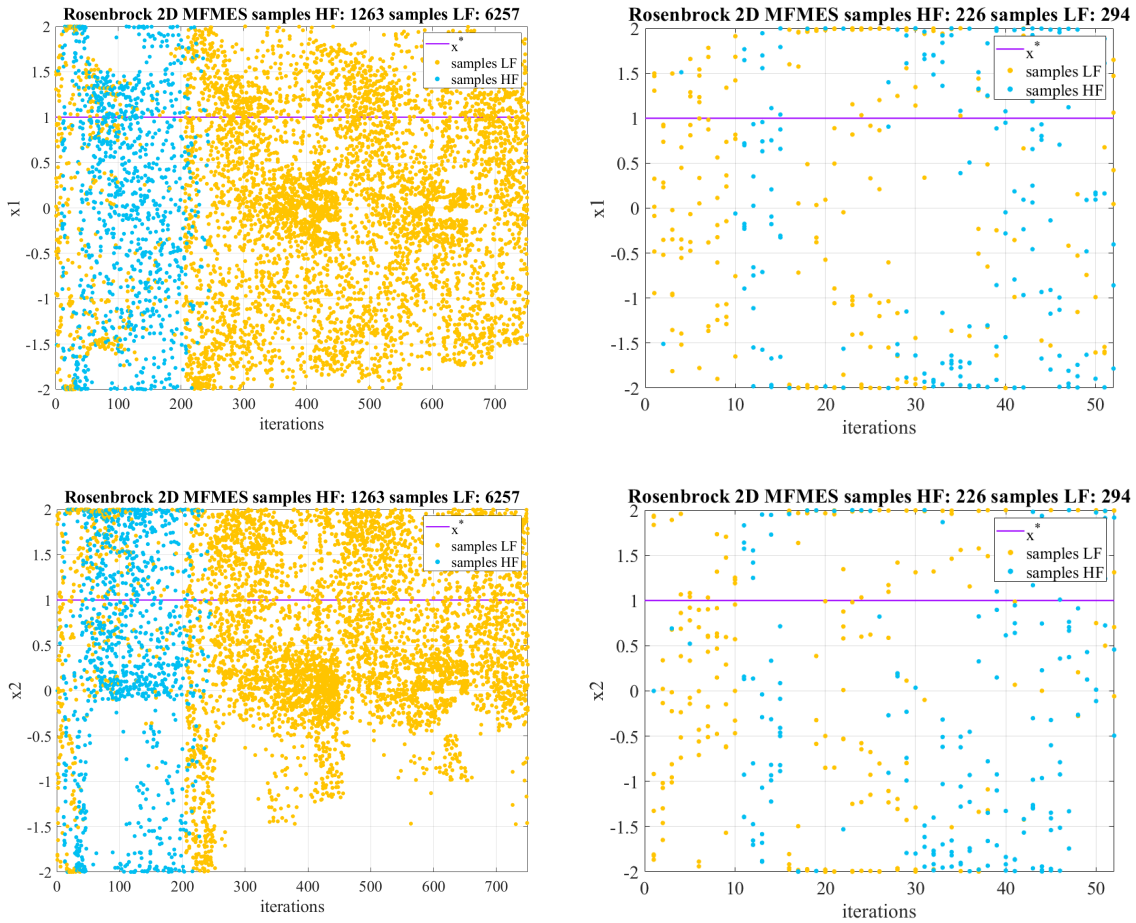
focuses only on the usage of the low fidelity and a repetition of the sampling strategy can be found, as seen with the MFEI. As for MFPI, even in this case, a thickness of points is located near x=(0,0). Also a cluster of points on the extremes can be found. Max-value Entropy Search takes about 52 iterations to catch the coordinates of the optimum, by means of 226 HF samples and 294 LF ones.



(a) Complete

(b) Until the optimum

Figure 5.15: Distribution of MFPI samples over x.



(a) Complete (b) Until the optimum

Figure 5.16: Distribution of MFMES samples over x.

5.4 Rosenbrock 5D

Extending the corresponding unimodal 2D function to higher dimensionality, converts the function in a multimodal one. This happens for $4 \leq D \leq 7$, where there is a local minimum located near $\boldsymbol{x} = (1, 1, ..., 1)$. Generally, by increasing further the dimensionality of the problem $D \geq 3$, more local minima can be found, but a theoretical analysis to evaluate them, has not yet been provided [95]. However, an analytical method to accurately approximate these values is experimentally implemented by Shang and Qiu [95].

Function	Dimension	Budget	HF cost	LF cost	HF samples	LF samples
Rosenbrock	5	500	1	0.1	15	30

Table 5.4: Rosenbrock 5D experiment setup

5.4.1 Results

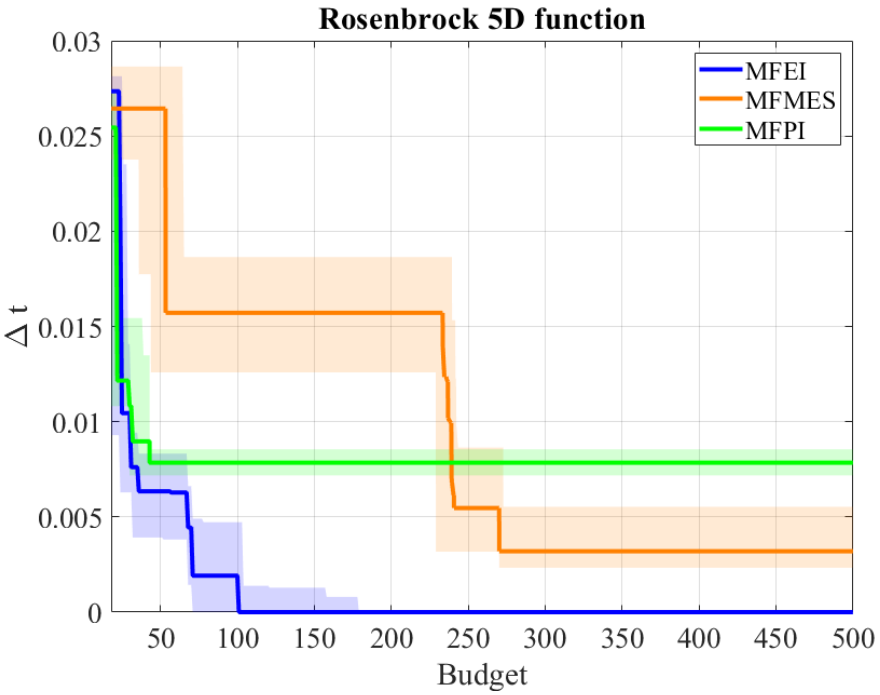


Figure 5.17: Comparison of MFEI, MFMES and MFPI on Rosenbrock 5D function.

In this case, only five tests have been run because of limited resources. Furthermore, even if Figure 5.17 is depicted till the end of budget, the following figures about the x spreading, are narrowed to the optimum achievement, because the corresponding total figures do not provide useful information. The Multifidelity Probability of Improvement initially is able to reduce more rapidly the error Δ_t than the other methods, but from budget 25, the Expected Improvement becomes more efficient and, indeed, it reaches the minimization of the error with a budget of 101. It is the only method that reaches the best residual error minimization, thus demonstrating that, even if the surrogate may loss efficiency with dimensionality, it still remains

efficient when guided by the appropriate acquisition function. Whereas, the MFPI, cannot improve beyond a residual error of 0.0078 that achieves at budget 43. On the other hand, MFMES starts with the worst behaviour, dealing with a greater confidence interval, and stops with a residual Δ_t of 0.003, that is a lower value than that obtained by the MFPI. MFEI method starts with an initial sampling of HF points, then, from the 6-th to the 44-th run, adds also LF points, but beyond the 45-th iteration, it is possible to observe a greater spread of high-fidelity patterns especially towards the zero coordinates. However, towards x^* there is not a slightly

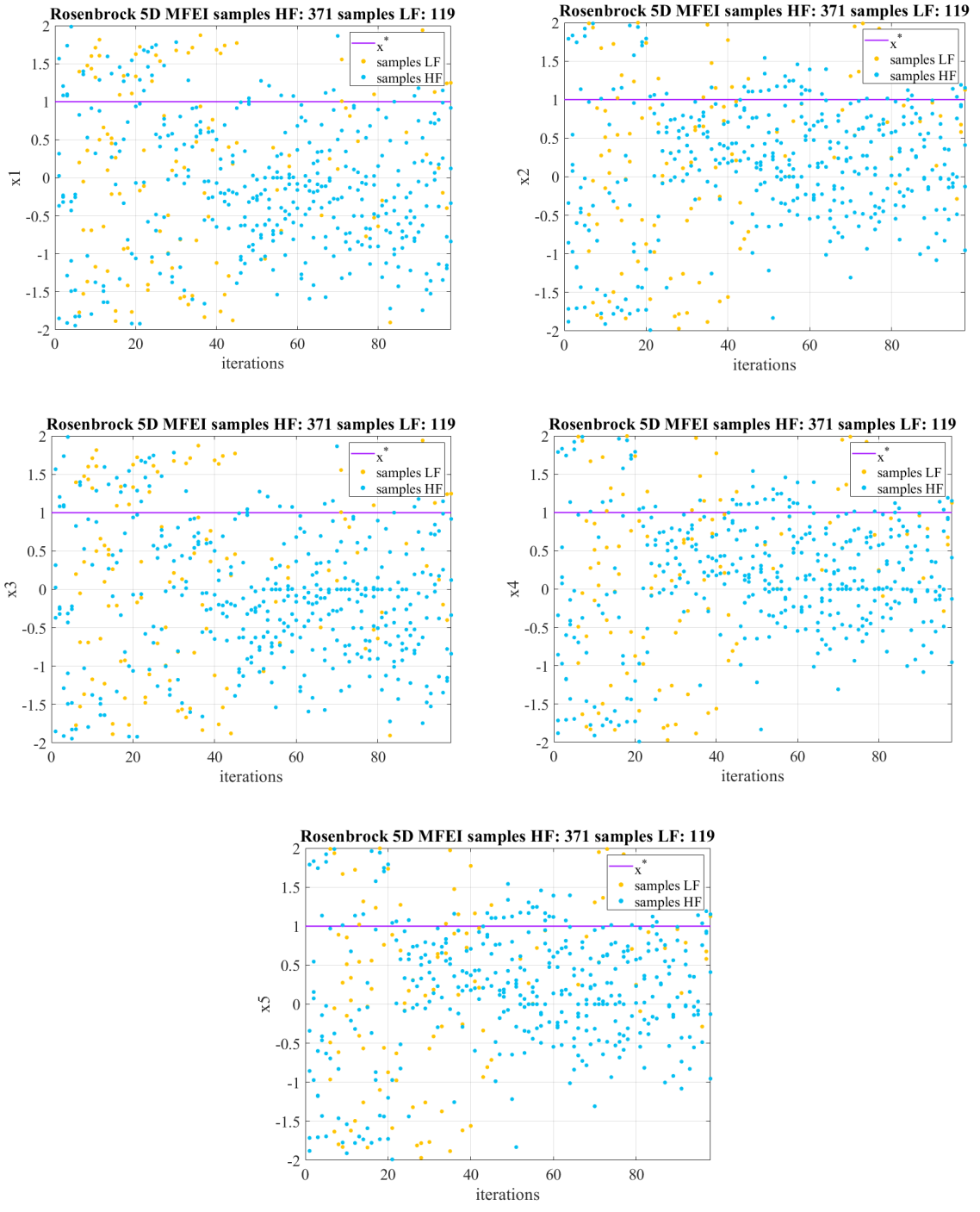


Figure 5.18: Distribution of MFEI samples over x.

cluster of points: it means that few tests were able to achieve the global minimum.

For the first iterations, MFPI prefers to exploit the space of design by using the high-fidelity model, then a greater amount of low-fidelity samples is employed till the iteration number 44. Later, more high-fidelity calls are present, with a tendency to sample near the zero coordinate. Also in this case, not all the tests reach the minimum, but it is possible to notice that it uses a smaller number of total samples than the MFEI, to achieve its optimum.

MFMES results the method that uses the greatest amount of total samples, to dis-

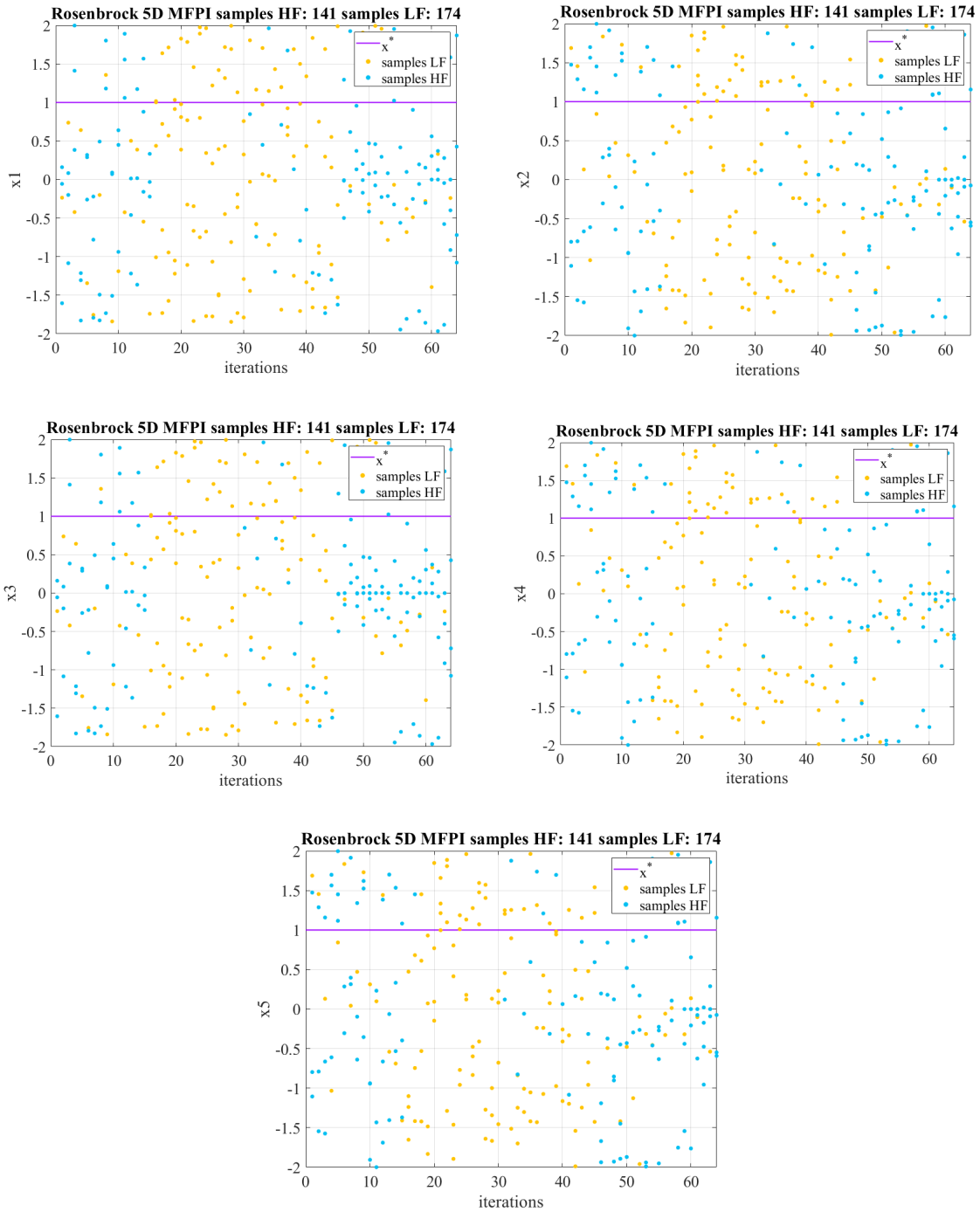


Figure 5.19: Distribution of MFPI samples over x.

cover the space of design. It employs from 7 to 10 times more points than the other methods. The first iterations widely use the low-fidelity model and a clustering of points can be found towards the extremes of the domain. At around the 400-th run a net increase of high-fidelity points is placed, with still a greater concentration towards the extremes. Even in this case, not all the tests reach the optimum.

Figure 5.20: Distribution of MFMES samples over x.

5.5 Shifted-Rotated Rastrigin function

This function is selected as benchmark for its marked multimodal behaviour. It is useful to approximate real problems, such as aeroelastic flutter. The function is shifted, to change the position of the minimum, and rotated, to change its properties within the variable domain. It is defined as:

$$f_1(\boldsymbol{z}) = \sum_{i=1}^{D}(z_i^2 + 1 - \cos(10\pi z_i))$$

where

$$\boldsymbol{z} = R(\theta)(\boldsymbol{x} - \boldsymbol{x}^*) \quad \text{with} \quad R(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

where R is the rotation matrix in a 2D framework, but can be extended to nD dimensions. The variables range within the interval $\boldsymbol{x} = [-0.1, 0.2]^D$, the rotation angle $\theta = 0.2$. The optimum is $f(\boldsymbol{x}^*) = 0$ at $\boldsymbol{x}^* = (0.1, \ldots, 0.1)$. Due to difficulties found in MFMES computation, because of the ill conditioning of the problem, a medium fidelity is selected to replace the lowest one. Hence, the cost to call this function, with respect to the HF, is set to 0.5, instead of the 0.0039 value proposed by the guidelines of the international group of work, and ϕ_2 is increased from 2500 (worst approximation function) to 5000. Hence, the medium-fidelity function is expressed as:

$$f_m(\boldsymbol{z}, \phi_m) = f_1(\boldsymbol{z}) + e_r(\boldsymbol{z}, \phi_m) \qquad m = 1, 2$$

with

$$\phi_1 = 10000$$

$$\phi_2 = 5000$$

$$e_r(\boldsymbol{z}, \phi_m) = \sum_{i=1}^{D} a(\phi_m)\cos^2(w(\phi_m)z_i + b(\phi_m) + \pi)$$

$$a(\phi_m) = \Theta(\phi_m)$$

$$w(\phi_m) = 10\pi\Theta(\phi_m)$$

$$b(\phi_m) = 0.5\pi\Theta(\phi_m)$$

$$\Theta(\phi_m) = 1 - 0.0001\phi_m.$$

It is the formulation proposed by [96], where $e_r(\boldsymbol{z}, \phi_m)$ is the resolution error.

Function	Dimension	Budget	HF cost	LF cost	HF samples	LF samples
Rastrigin	2	200	1	0.5	10	20

Table 5.5: Shifted-Rotated Rastrigin experiment setup

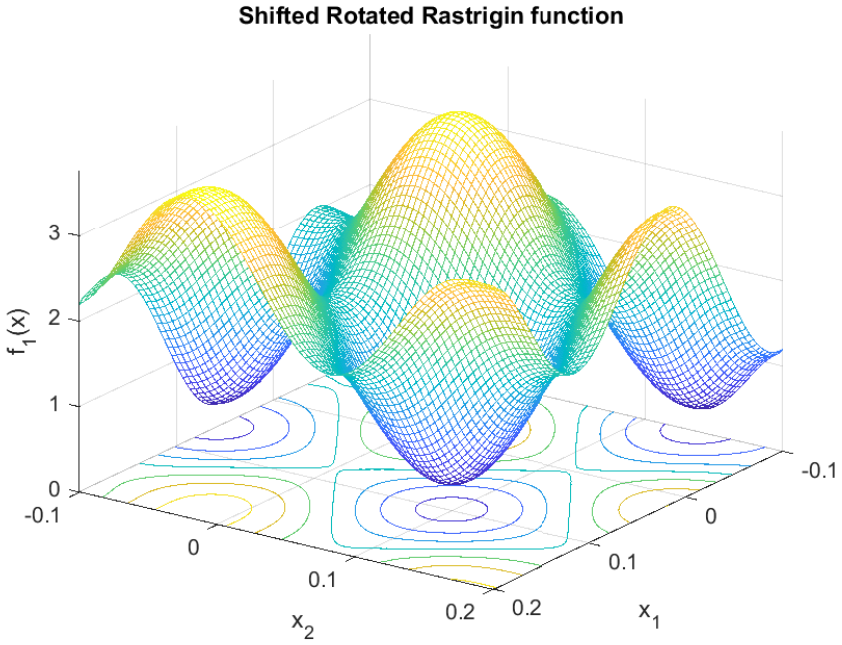


Figure 5.21: Shifted-Rotated Rastrigin high-fidelity function

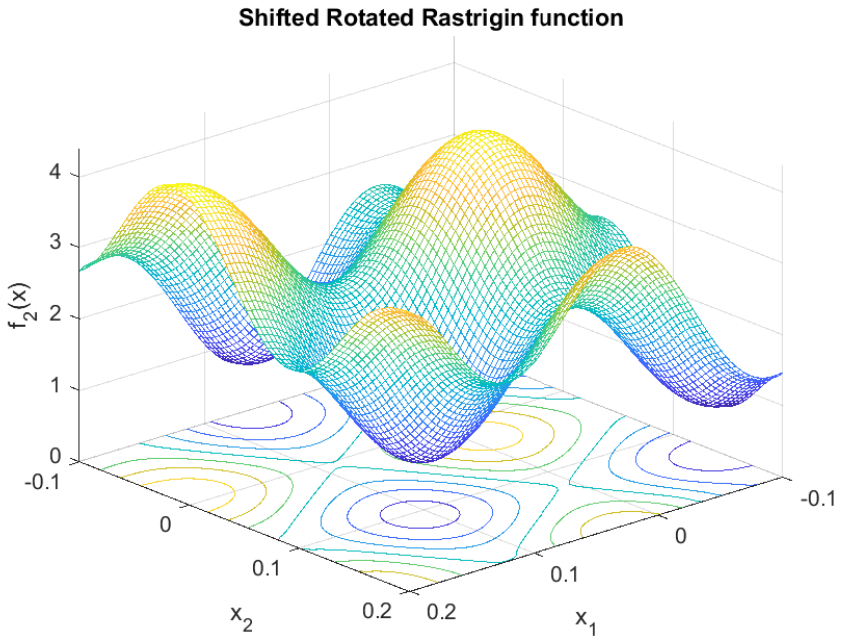


Figure 5.22: Shifted-Rotated Rastrigin medium fidelity function

5.5.1 Results

The behaviour of the methods is comparable with that previously seen for the sinusoidal squared function. Therefore, for what concerns the Expected Improvement and the Probability of Improvement functions, the median efficiently decreases, by steps, spending until the 34% of the budget.

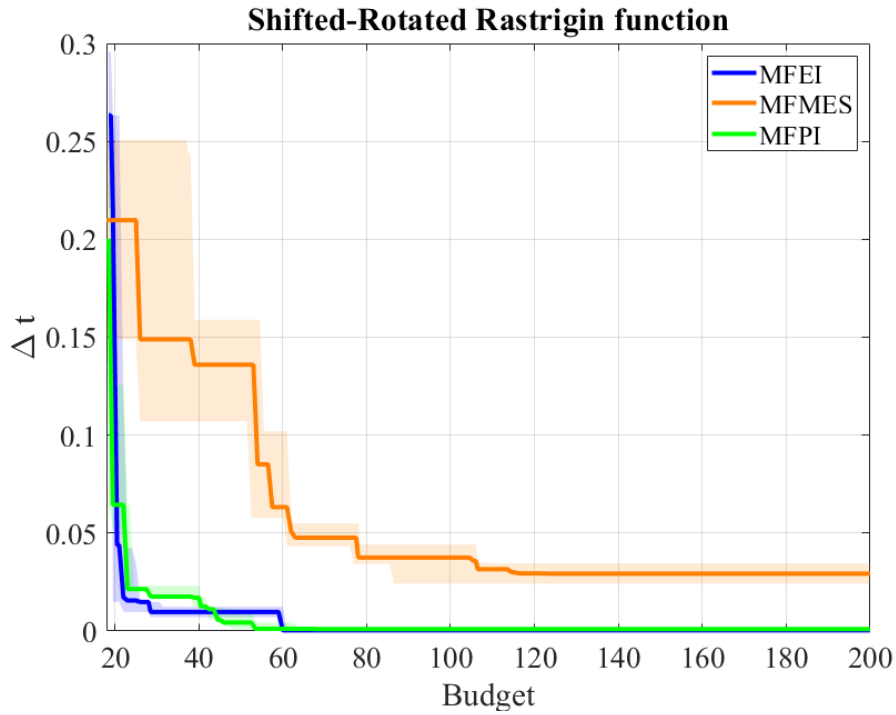
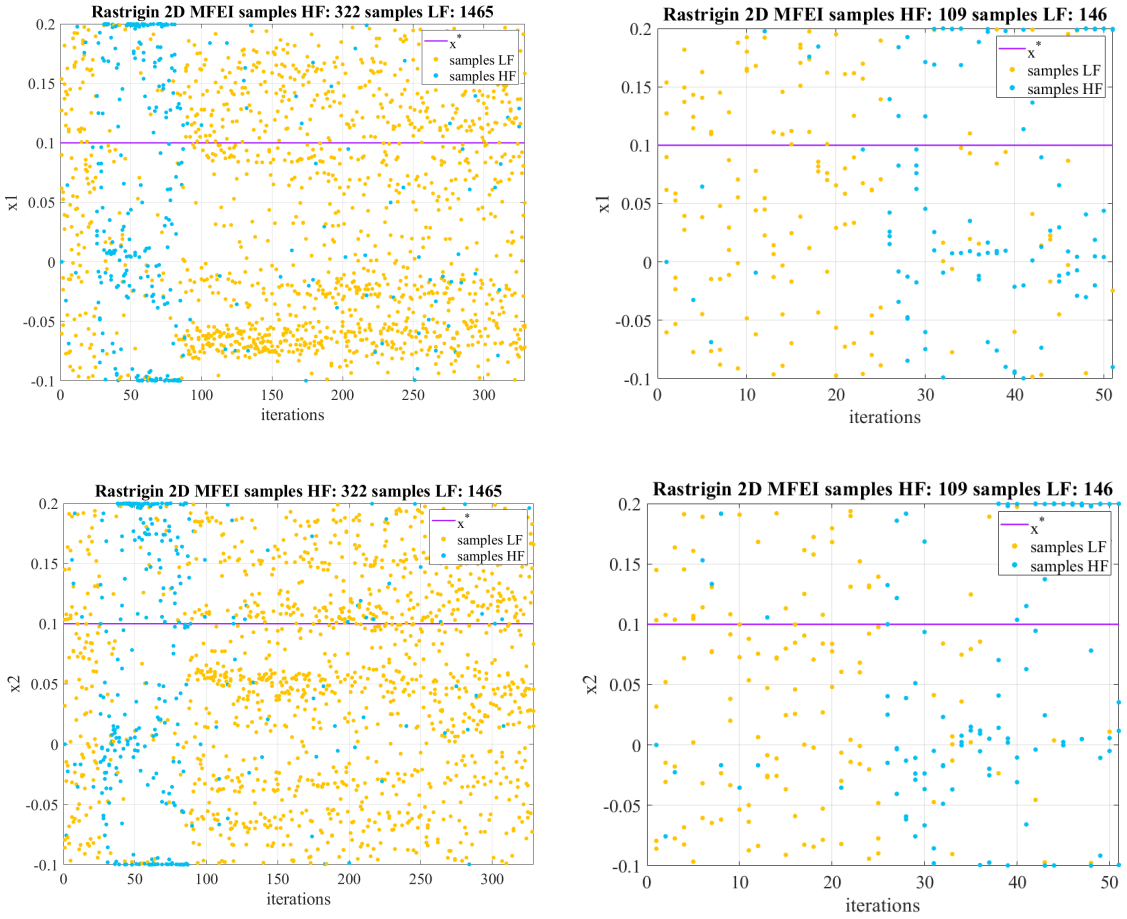


Figure 5.23: Comparison of MFEI, MFMES and MFPI on Shifted-Rotated Rastrigin function.

The initial speed of convergence of the MFPI, is then recovered by the MFEI. The last one, reaches zero with a budget equal to 60, in contrast with the MFPI, that converges with a value of budget equal to 68. MFMES cannot reduce the error more than 0.029, that is achieved with a budget of 124, but it is possible to notice that there are also some tests that were able to reach a minor error: the shaded area of the confidence interval extend beyond 0.013. Hence, the median remains constant because, as the iterations proceed, the acquisition function tends to call especially LF samples to save budget.
MFEI uses about 51 iterations, 109 HF and 146 LF calls, to find the optimum location, generally using the low-fidelity function. Then, because the optimization process has still budget to employ, high fidelity samples are spread over the domain, especially at the extremes. This is always attributable to the surrogate accuracy. From the 90-th run till the end, especially LF patterns are used. After the convergence, a slight cluster of point is visible towards the local minimum in x=(-0.05,0.05).
MFPI, does not converge until the 62-th iteration is passed. The method employs around 166 and 144 high- and low-fidelity points. It is possible to see a cluster of LF and HF samples towards x=(0,0), in which there is a local minimum, and towards the extremes of the search space. MFEI and MFPI uses nearly an equal number of total samples to discover the space of design, with a ratio between HF and LF patterns of about 1:4 and 1:3, respectively. However, to find the optimum, the MFPI uses more the high-fidelity model than the MFEI.
Also MFMES, tends to collect HF samples towards the local minimum, and moves close to the global optimum at the 139-th run, by using 331 and 364 high- and low-fidelity patterns. The Max-value Entropy Search uses a total of about 700 points less than the other two methods, with a difference between low- and high-fidelity samples of about 100 points. Hence, with a total of around 1095 points selected,

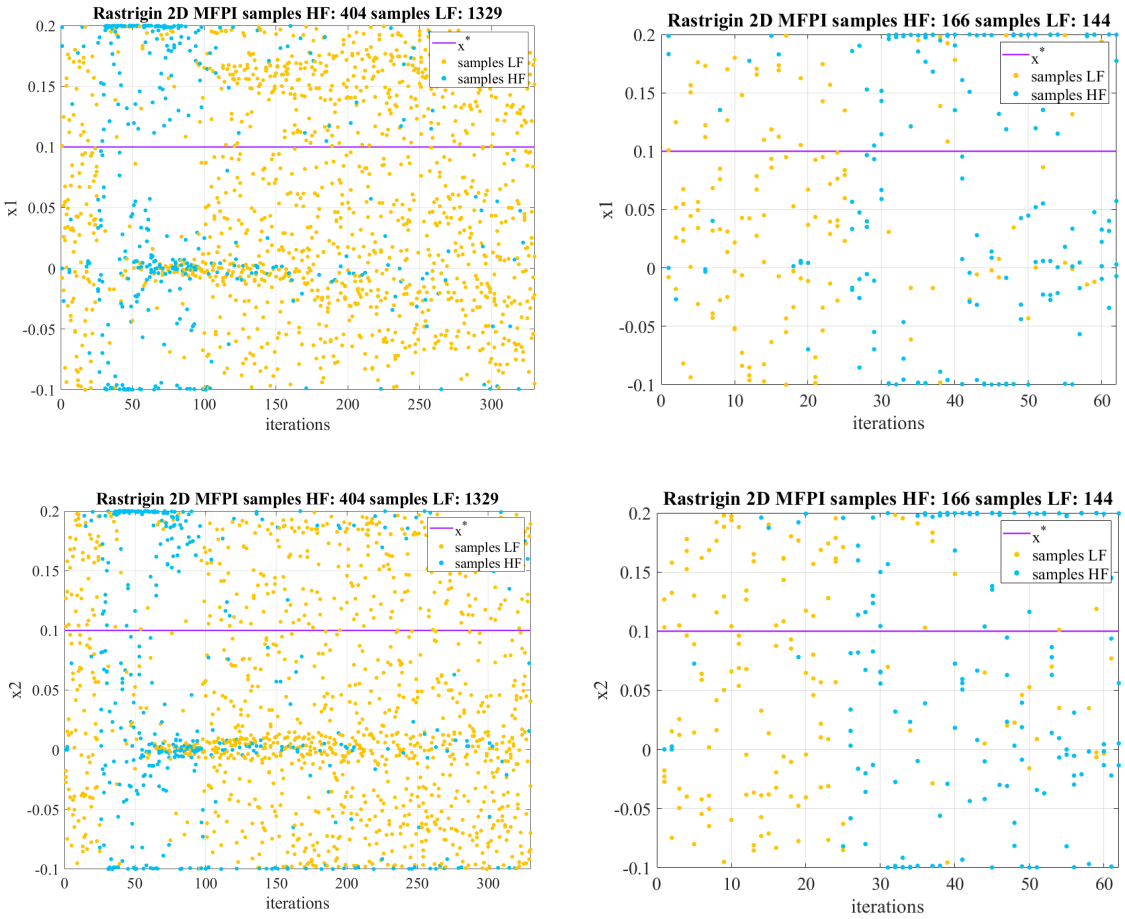
about the 63% of them is spent to move close to the optimum. Then the method tends to cover the upper extreme of the domain especially with the high-fidelity model. Later it searches in the lower part of the space of design, by using especially the low-fidelity function.



(a) Complete

(b) Until the optimum

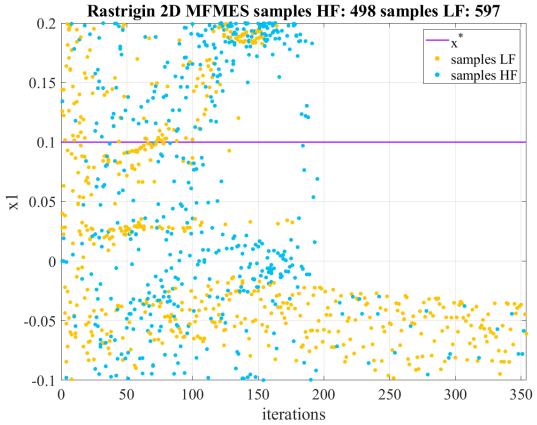
Figure 5.24: Distribution of MFEI samples over x.

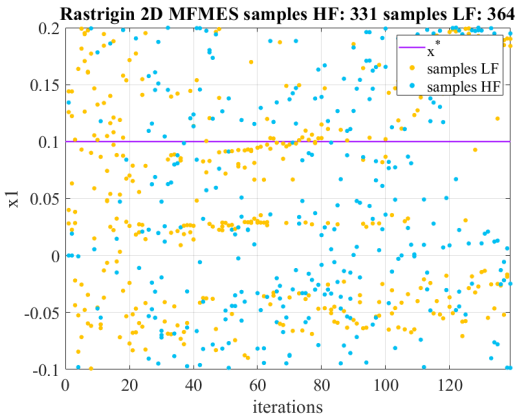


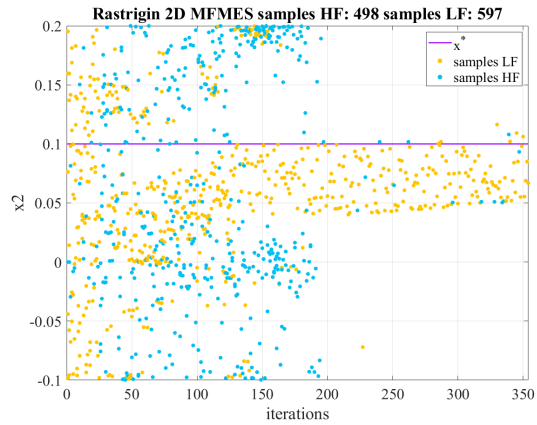
(a) Complete

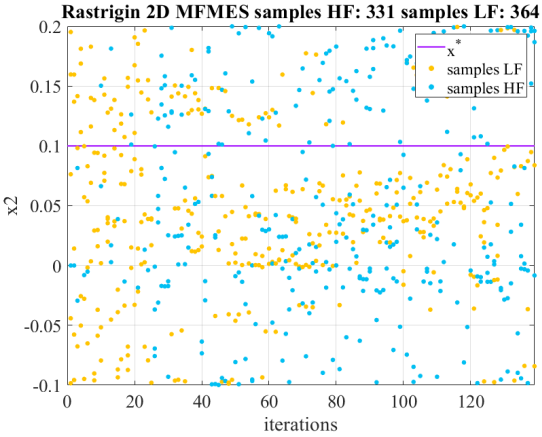
(b) Until the optimum

Figure 5.25: Distribution of MFPI samples over x.









(a) Complete

(b) Until the optimum

Figure 5.26: Distribution of MFMES samples over x.

5.6 Clark and Bae 1D function

This is a heterogeneous and non-polynomial function defined on unit cubes in one dimensions (then, it will be defined also in two and three dimensions), in order to assess the effectiveness agglomeration of locally optimized surrogates. The domain is within [0,1] and, in one-dimensional space, the minimum is $f(\boldsymbol{x}^*) = -0.6250$ at $\boldsymbol{x}^* = 0.2753$. High fidelity function is taken from [25], whereas the low-fidelity version is obtained by scaling. Clark and Bae functions are useful to assess the goodness of the surrogate model. The two fidelity models are computed as:

$$f_1(\boldsymbol{x}) = \sin[30(\boldsymbol{x} - 0.9)^4]\cos[2(\boldsymbol{x} - 0.9)] + (x - 0.9)/2$$

$$f_1(\boldsymbol{x}) = (f_1 - 1.0 + \boldsymbol{x})/(1.0 + 0.25\boldsymbol{x}).$$

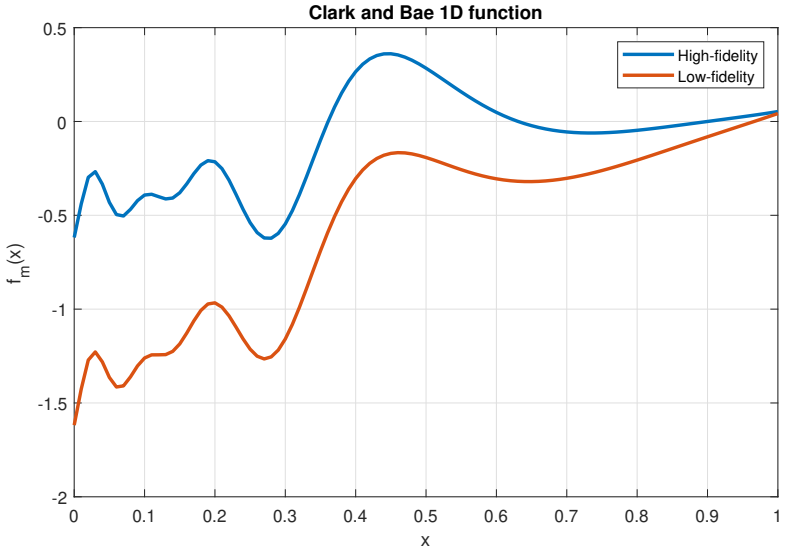


Figure 5.27: Clark and Bae 1D function

Function	Dimension	Budget	HF cost	LF cost	HF samples	LF samples
Clark and Bae	1	100	1	0.2	2	5

Table 5.6: Clark and Bae 1D experiment setup

5.6.1 Results

The results indicate better performances of the MFEI, in reducing the median, but from budget=44, the MFPI dominates achieving a residual error lower than that of MFEI. Max-value Entropy Search reduces the error more slowly and stops with a residue of one order of magnitude greater than the other methods.
The samples distribution computed by the MFEI, employs an initial set of LF samples, and then leads to the analysis of the design domain, by increasing the HF patterns till the 40-th run. The method converges after about 93 iterations, thanks

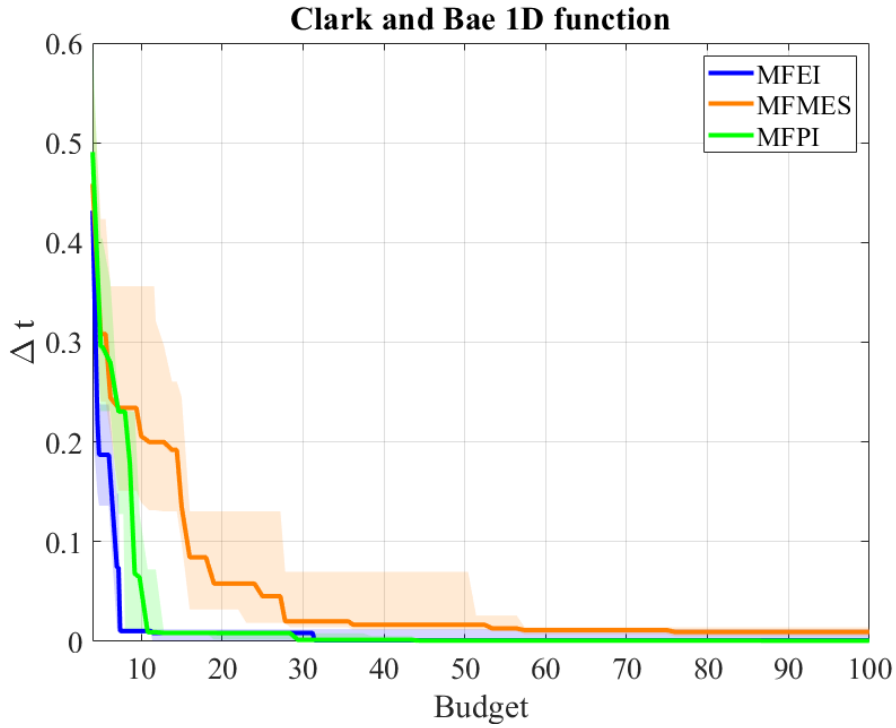
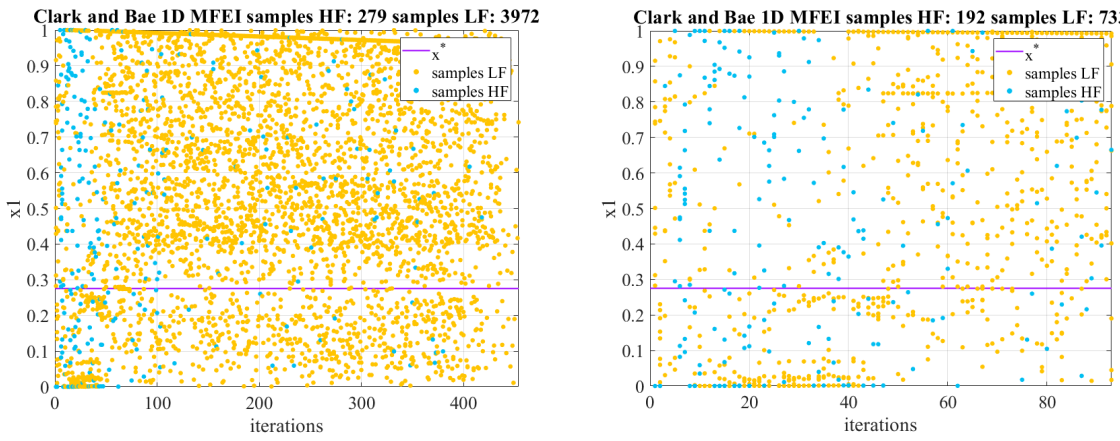


Figure 5.28: Comparison of MFEI, MFMES and MFPI on Clark and Bae 1D function.

to 192 HF samples and 733 LF ones. Hence, it is possible to see that after achieving the optimum, the algorithm does not search anymore towards x^*. A cluster of points towards the extremes of the domain can be found, also in the first iterations, then the samples are widely spread across all the domain by employing especially the LF.

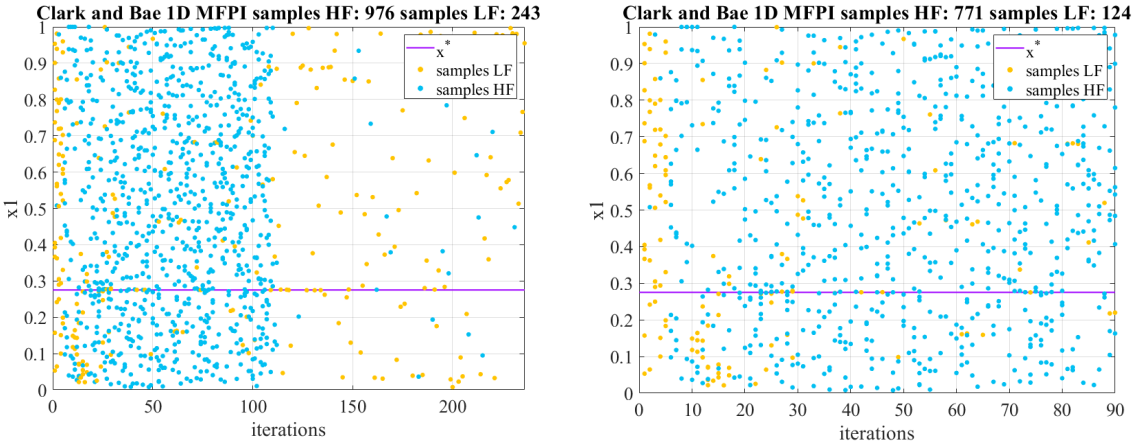
MFPI reveals a clear initial distinction between low- and high-fidelity functions calls. After seeding the first samples using the low-fidelity model and not finding a reasonable minimum, it focuses on the usage of high-fidelity till budget= 110. Then it looks around by allocating low-fidelity samples. About 90 iterations are required to catch the global optimum, with 771 HF and 124 LF points. It is the method that employs the greatest number of HF points, so the smallest amount of total samples.



(a) Complete (b) Until the optimum

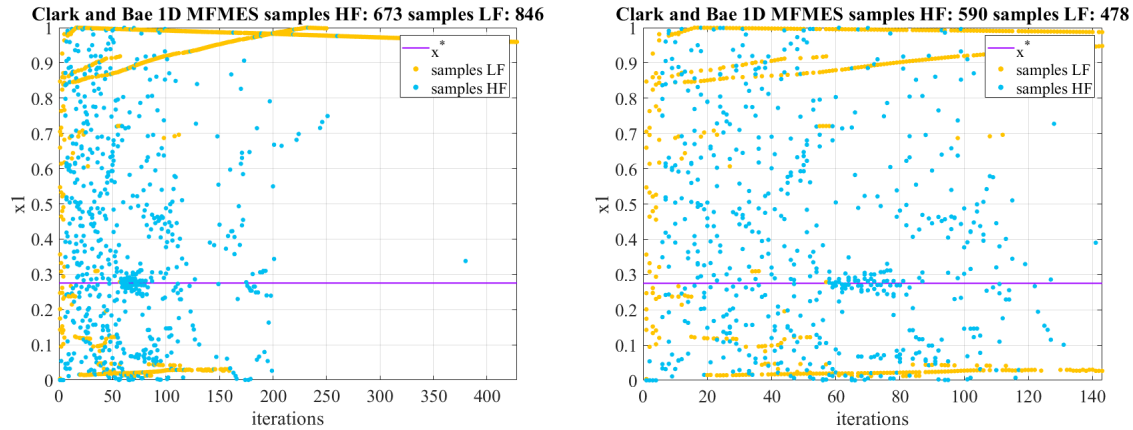
Figure 5.29: Distribution of MFEI samples over x.

MFMES finds the global minimum with 143 iterations and using a high amount of HF patterns: 590 samples with respect to the 478 LF. Then a great density of HF points is located towards x^*, within a budget that ranges from 60 to 80. Search paths towards the extremes can be seen, and are essentially made of LF patterns.



(a) Complete (b) Until the optimum

Figure 5.30: Distribution of MFPI samples over x.



(a) Complete (b) Until the optimum

Figure 5.31: Distribution of MFMES samples over x.

5.7 Clark and Bae 2D function

According with [25], in two-dimensional space, Clark and Bae high- and low-fidelity functions become respectively:

$$f_1(\boldsymbol{x}_1, \boldsymbol{x}_2) = \sin[21(\boldsymbol{x}_1 - 0.9)^4] \cos[2(\boldsymbol{x}_1 - 0.9)] + (\boldsymbol{x}_1 - 0.7)/2 + 2\boldsymbol{x}_2^2 \sin(\boldsymbol{x}_1 \boldsymbol{x}_2)$$

$$f_2(\boldsymbol{x}_1, \boldsymbol{x}_2) = (f_1 - 2.0 + \boldsymbol{x}_1 + \boldsymbol{x}_2)/(5.0 + 0.25\boldsymbol{x}_1 + 0.5\boldsymbol{x}_2).$$

The minimum is $f(\boldsymbol{x}^*) = -0.5627$ at $x_1 = 0.0395$ and an isominimum along x_2. Its presence inhibits the possibility to compute the error as Δ_t, thus in this case it is assessed just as Δ_f.

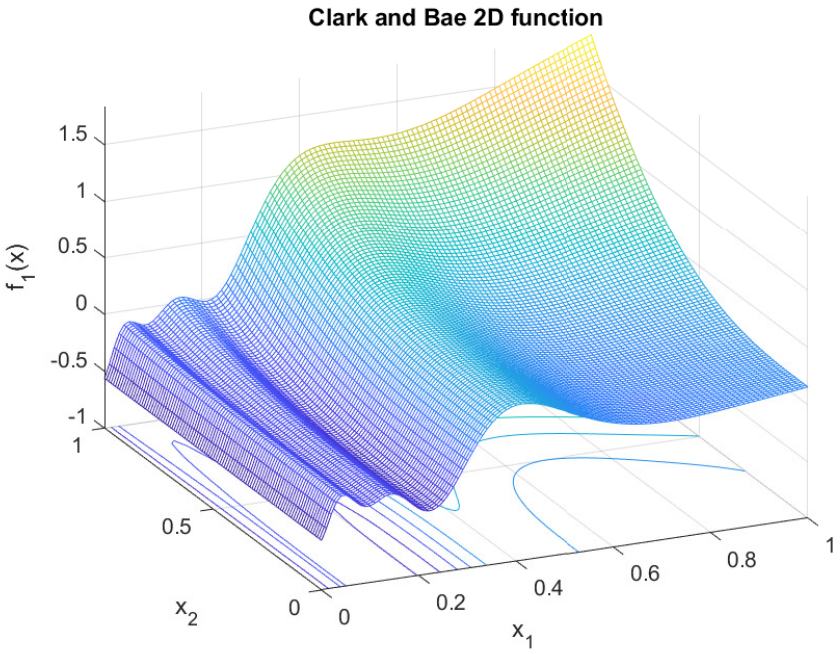


Figure 5.32: Clark and Bae 2D high-fidelity function

Function	Dimension	Budget	HF cost	LF cost	HF samples	LF samples
Clark and Bae	2	200	1	0.2	5	10

Table 5.7: Clark and Bae 2D experiment setup

5.7.1 Results

Being a projection to higher dimensionality of the Clark and Bae 1D, this function tends to be really flat, thus providing a strong ill conditioning of the covariance matrix, that does not make possible to execute the Cholesky factorization to build the Gaussian Process: therefore, it is possible to notice that, as the number of iterations increases, the thickness of points reduces, because more tests stop before reaching the end of budget. In this case, the MFPI results the method that reduces the function error more rapidly than the others, however the minimization of the

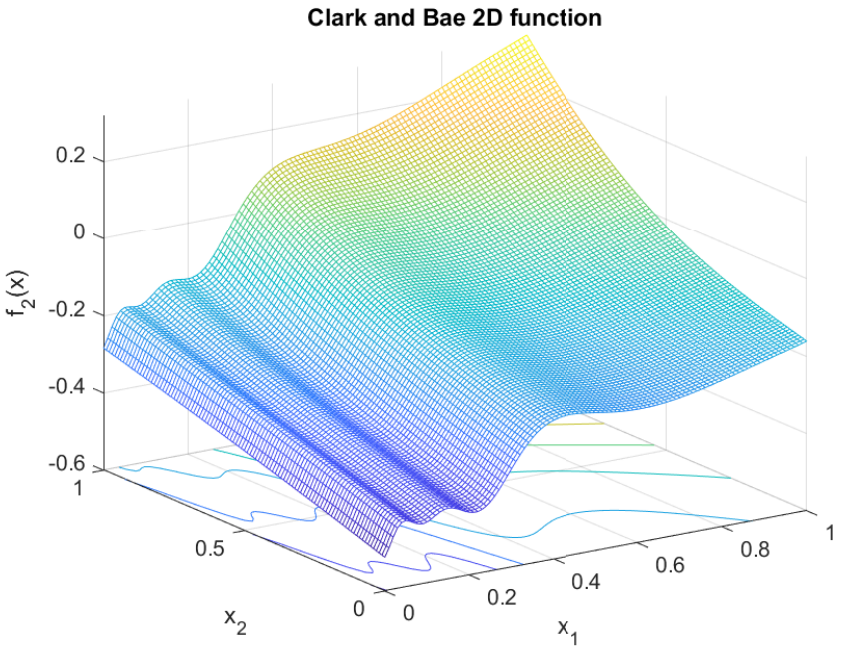


Figure 5.33: Clark and Bae 2D low-fidelity function

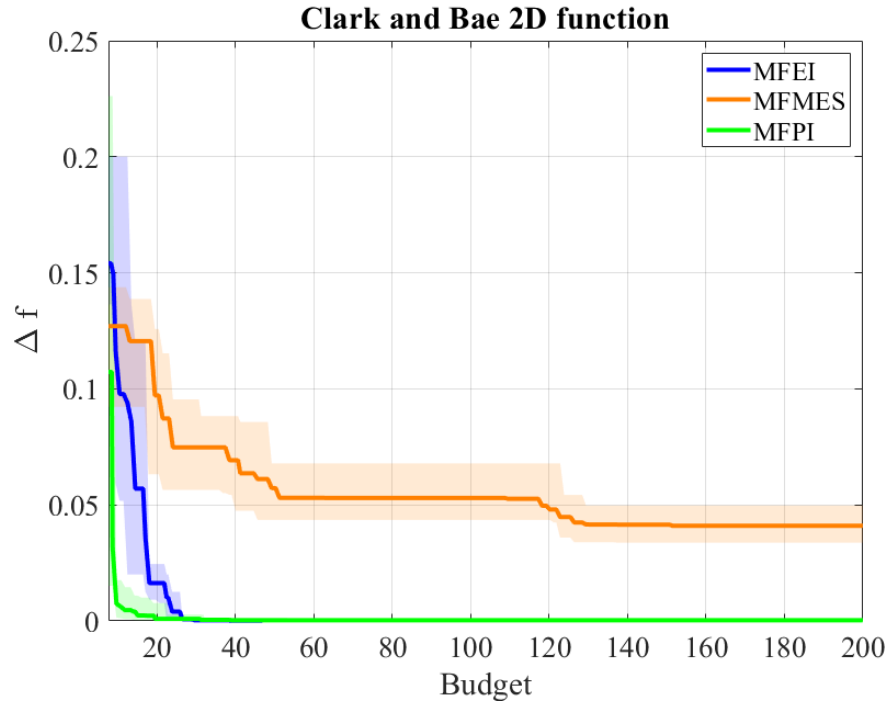
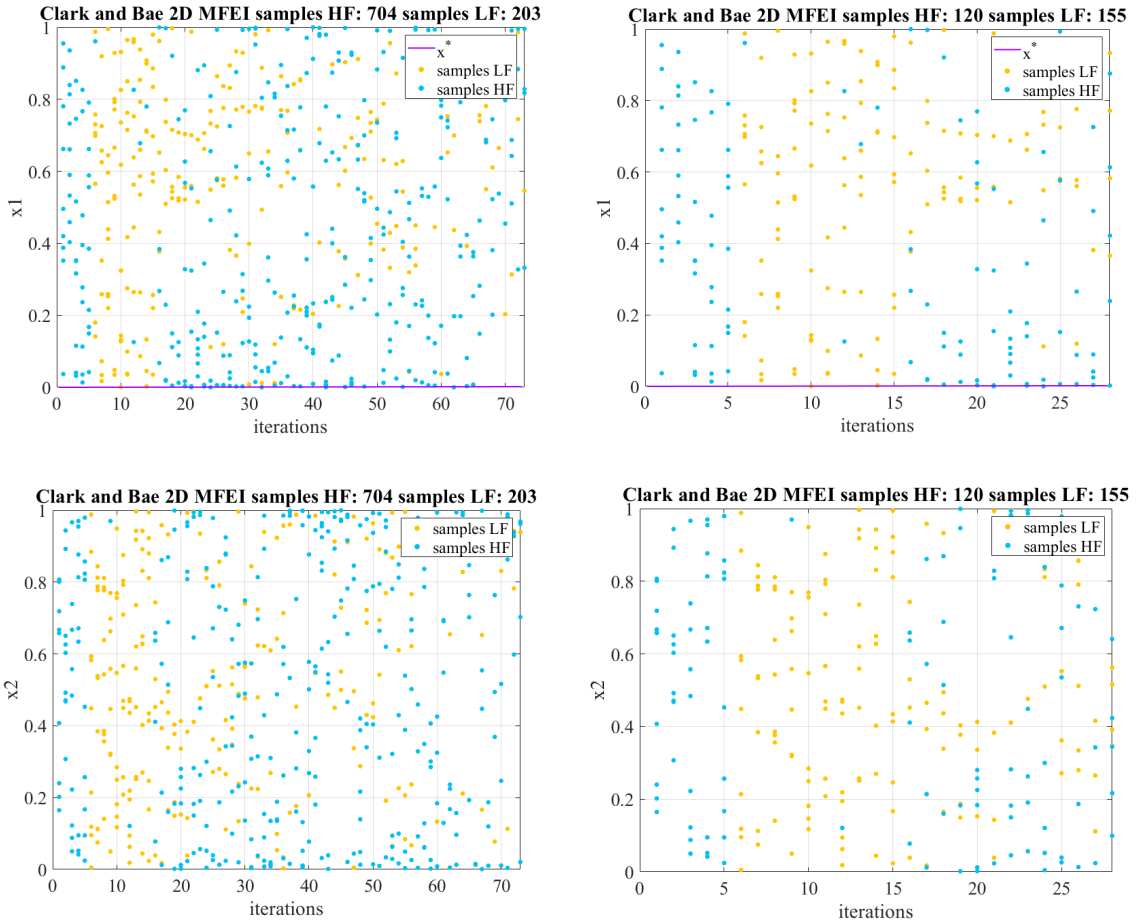


Figure 5.34: Comparison of MFEI, MFMES and MFPI on Clark and Bae 2D function.

median is reached with a budget of 53, in contrast with the budget spent, for the same purpose, by the MFEI (around 38). MFMES, whereas, shows a slower reduction of the error and stops this tendency, when achieves a budget of 152.
The Expected Improvement employs, in mean, 28 iterations looking for the optimum, with 120 HF and 155 LF patterns. It generally uses a higher amount of high fidelity calls: 704 with respect to the 203 calls of the low-fidelity function. However,

to achieve the convergence, more LF, than HF points, are employed. It is possible to notice a light tendency to spread points over the extremes of the domain by using the high-fidelity model.

The Probability of Improvement uses 74 iterations to find the global optimum, with



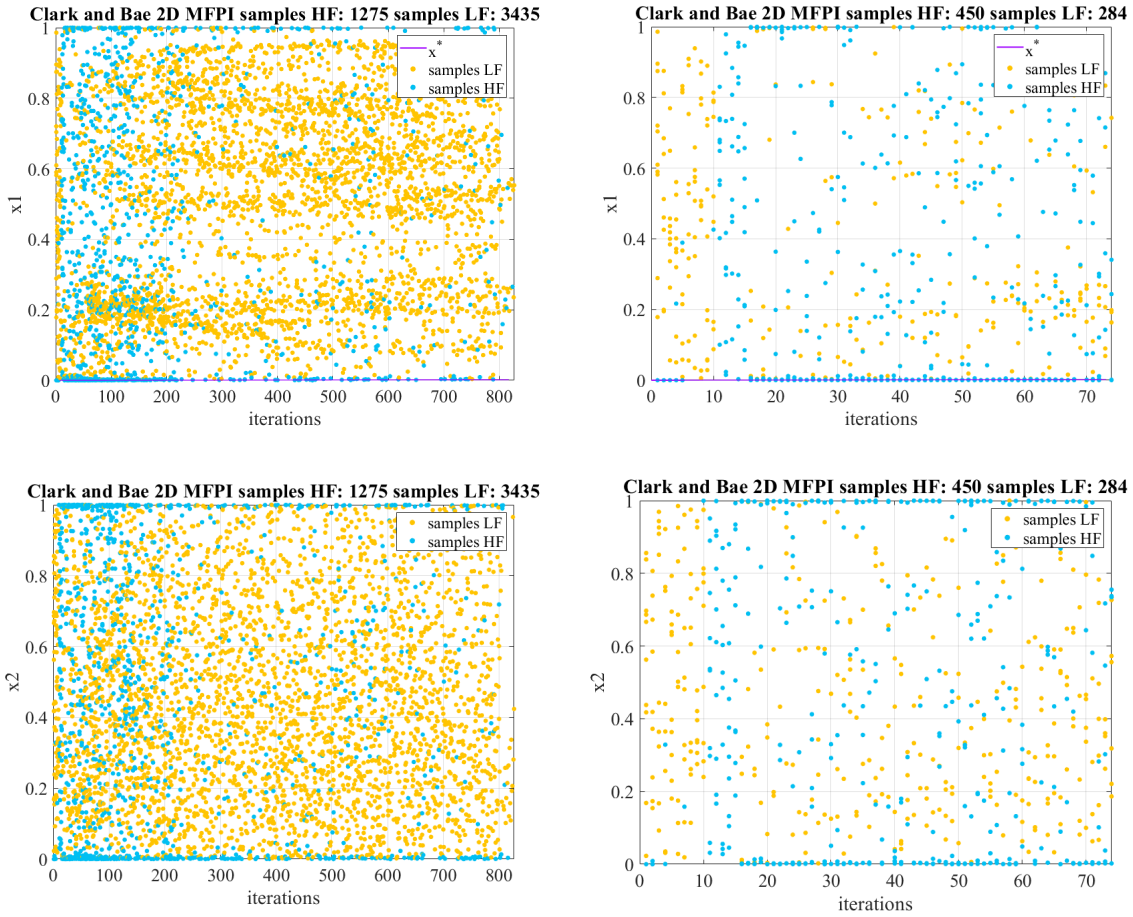
(a) Complete

(b) Until the optimum

Figure 5.35: Distribution of MFEI samples over x.

450 and 284 high- and low- fidelity calls. In this case, a greater thickening of points, than what has been seen for MFEI, is located towards the extremes of the space of design. Once x_1 finds the zero, any other point selected along the x_2 coordinate is acceptable, because of the isominimum presence in this function. Hence, along x_2, is it possible to observe a quite uniform sampling all over the space of design, especially using LF points. The reduced thickening of points as the iterations increase, is due to the ill conditioning of the covariance matrix that not allows to take all the tests till the end of the budget.

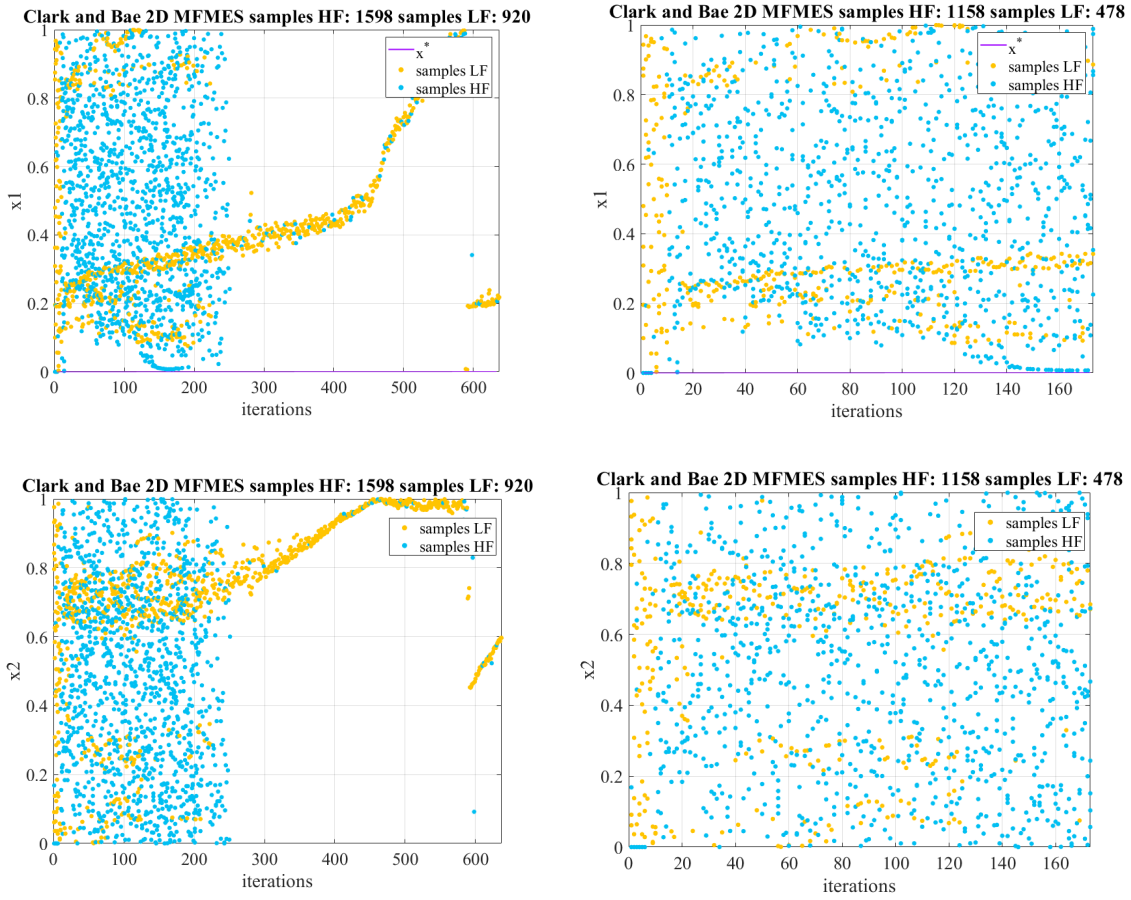
The Max-value Entropy Search, uses 173 iterations, 1158 HF and 478 LF samples for the optimum. The first iterations tend to call especially the low-fidelity function, but after achieving the 10-run, an increasing amount of HF samples are adopted. Therefore, around 678 HF extra points are used with respect to the LF ones. Sampling seems to cluster towards a local minimum of the high-fidelity function, approximately at $x = (0.24, 0.7)$, and then, it tends to follow a trajectory, especially of low-fidelity points, towards the upper limit of the domain.



(a) Complete

(b) Until the optimum

Figure 5.36: Distribution of MFPI samples over x.



(a) Complete

(b) Until the optimum

Figure 5.37: Distribution of MFMES samples over x.

5.8 Clark and Bae 3D function

An extension to three dimensions of the previous Clark and Bae functions is computed following the formulation below:

$$f_1(\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3) = \sin[21(\boldsymbol{x}_1 - 0.9)^4] \cos[2(\boldsymbol{x}_1 - 0.9)] + (\boldsymbol{x}_1 - 0.7)/2 + \\ + 2\boldsymbol{x}_2^2 \sin(\boldsymbol{x}_1 \boldsymbol{x}_2) + 3\boldsymbol{x}_3^3 \sin(\boldsymbol{x}_1 \boldsymbol{x}_2 \boldsymbol{x}_3)$$

$$f_2(\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3) = (f_1 - 2.0 + \boldsymbol{x}_1 + \boldsymbol{x}_2 + \boldsymbol{x}_3)/(5.0 + 0.25\boldsymbol{x}_1 + 0.5\boldsymbol{x}_2 - 0.75\boldsymbol{x}_3).$$

The minimum is $f(\boldsymbol{x}^*) = -0.5627$. Therefore, also in this case, the presence of an isominimum leads to compute the error as Δ_f.

Function	Dimension	Budget	HF cost	LF cost	HF samples	LF samples
Clark and Bae	3	300	1	0.2	10	20

Table 5.8: Clark and Bae 3D experiment setup

5.8.1 Results

The MFPI allows to reduce more rapidly the error, but from budget 31, it loses this great advantage and is passed by the MFEI, which minimizes the error with a budget of 33.6, with respect to its counterpart that spends a budget of 103. The MFMES requires always more budget, but cannot reach the annulment of the error: it stops at 216, with a residual error of 0.025 and a confidence interval quite large. MFEI adopts about 25 runs to achieve the optimum, by means of 107 HF and

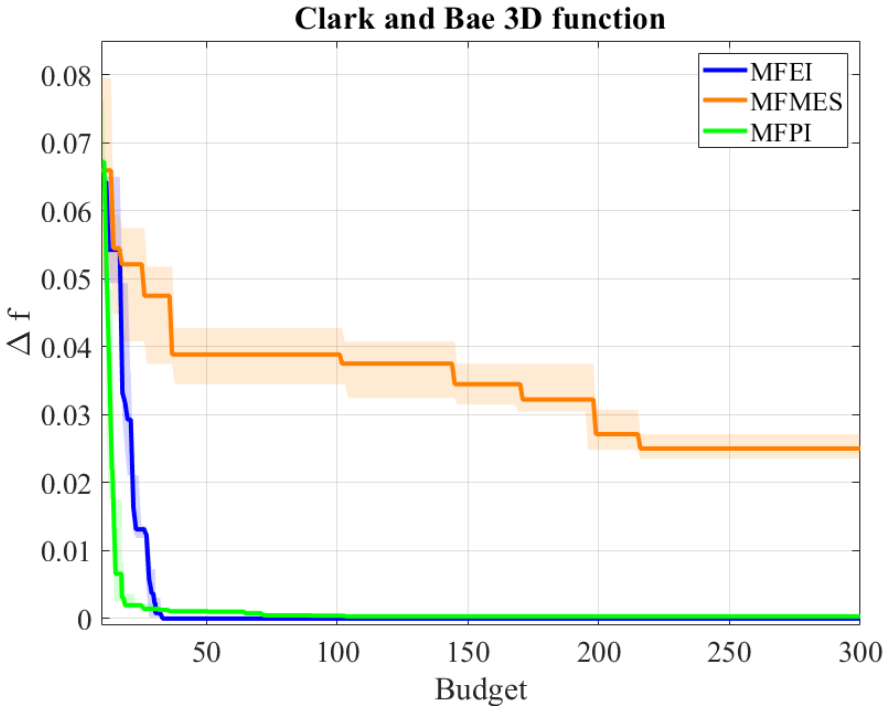
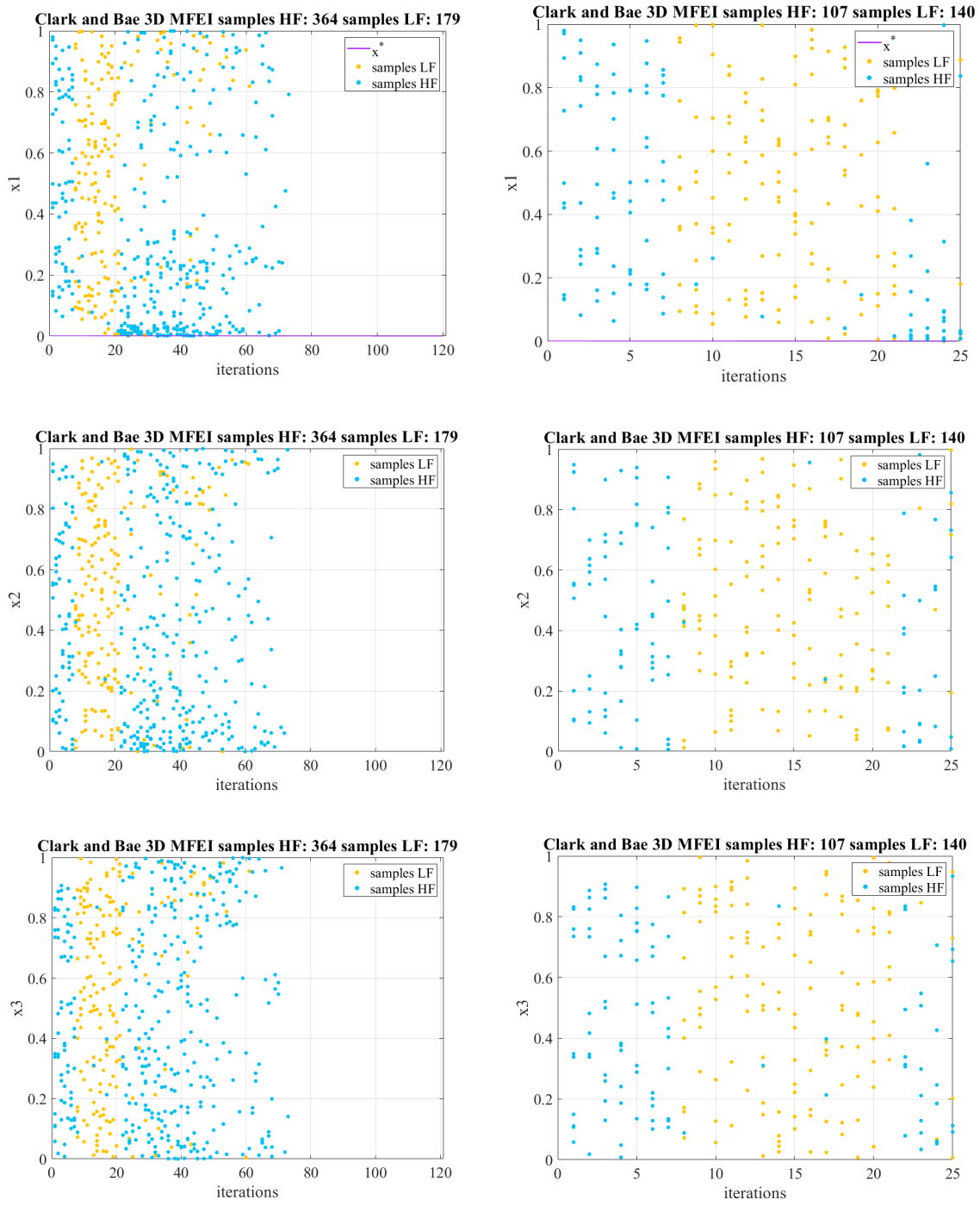


Figure 5.38: Comparison of MFEI, MFMES and MFPI on Clark and Bae 3D function.

140 LF samples. Indeed, a cluster of point is visible towards the zero of the x_1 coordinate, while a quite uniform diffusion of points is showed for the remaining two coordinates (isominimum). The method starts by using HF patterns, then between the 10-th and the 20-th iteration prefers the LF, but from this point to the end, adopts essentially high-fidelity points. The reduction of points from the 60-th run, is due to the impossibility to spend all the budget available, for the same problem of ill conditioning described for the Clark and Bae 2D function.

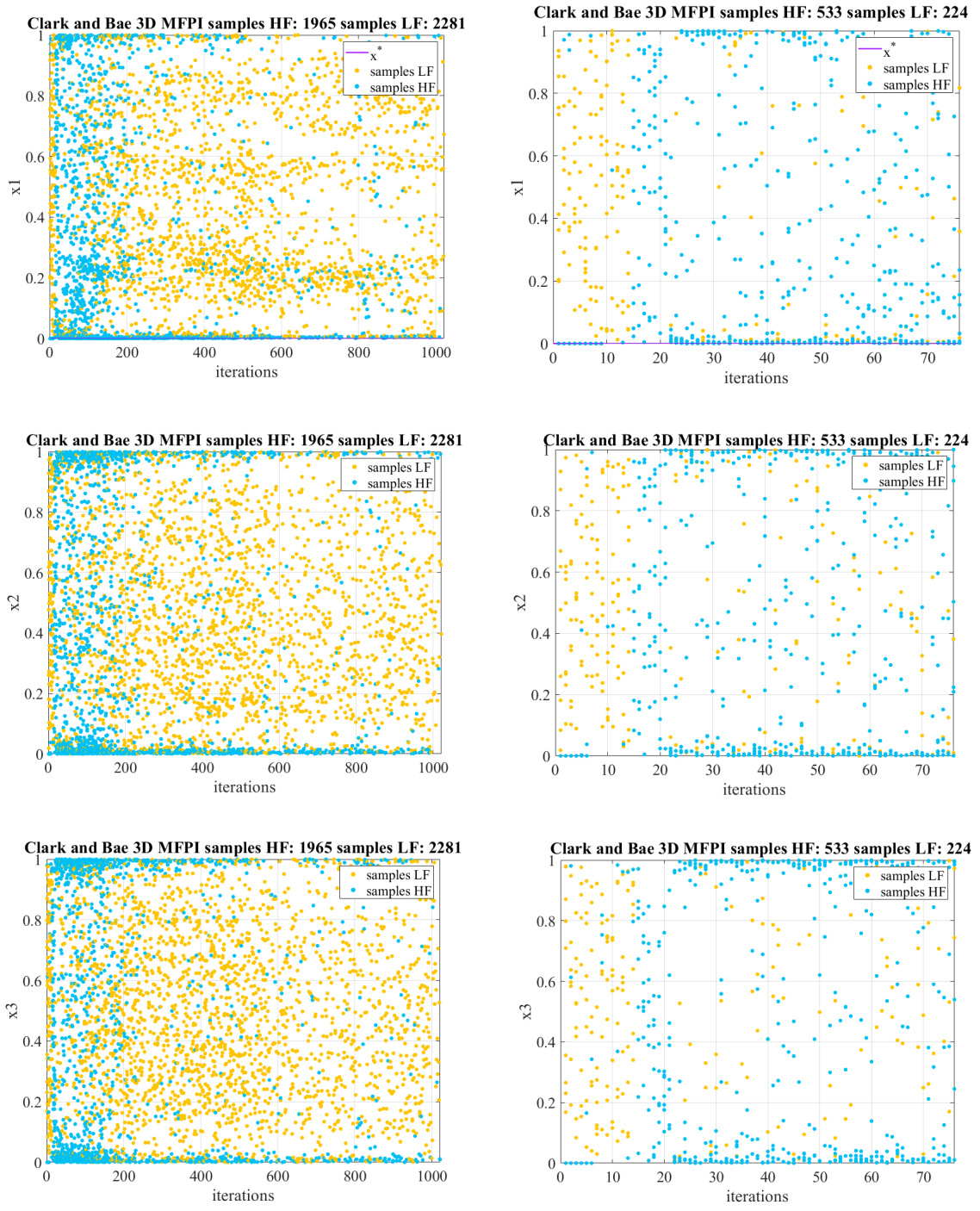


(a) Complete

(b) Until the optimum

Figure 5.39: Distribution of MFEI samples over x.

Also MFPI shows a behaviour that tends to sample initially by using the low-fidelity, and then clustering especially HF points towards the extremes, whereas, low-fidelity is called for spreading points over the remaining domain. It employs 76 runs to reach the target with 533 HF and 224 LF patterns. It uses a total number of samples nearly eight times greater than that used by MFEI. Furthermore, it needs more high-fidelity calls to achieve the optimum, with respect to the Expected Improvement.
For what concerns the MFMES, the optimum is reached by means of 275 runs, 590

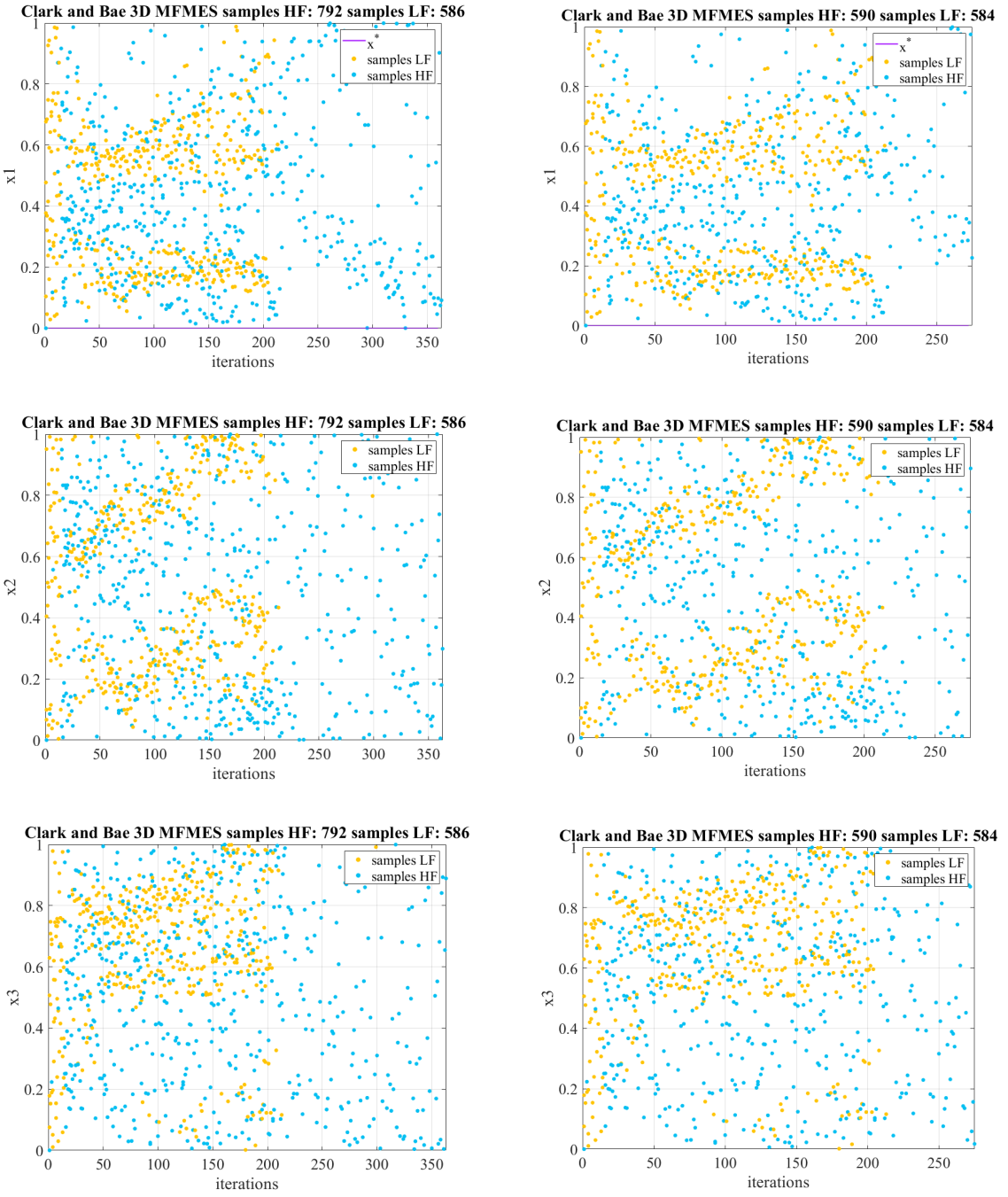


(a) Complete

(b) Until the optimum

Figure 5.40: Distribution of MFPI samples over x.

HF and 584 LF points. There is not a clear tendency to exploitation or exploration, indeed, it is important to highlight not only the inherent characteristic of the method to build a trade-off between the two approaches, but also the loss of accuracy of the surrogate with increasing dimensionality that encourages a wide search path throughout almost the entire domain. Therefore, until the 200-th run, a combination of high- and low- fidelity calls is showed, then the high-fidelity function is preferred.



(a) Complete

(b) Until the optimum

Figure 5.41: Distribution of MFMES samples over x.

5.9 Spring-Mass System 4D

The system is composed of two masses attached to each other by two springs. It is useful to evaluate the method behaviours with increasing dimensionality, indeed, it is trivial to add more masses and springs.
The masses m_1, m_2 are assumed to be concentrated at their center of gravity, so they are constant, while the mass of each spring is considered negligible. They follow the Hooke's law, therefore, k_1, k_2 denote the Hooke's constants. The springs return to their initial position, after compression and extension. The first and last spring are attached to fixed walls. The mass positions along the horizontal surface, in a frictionless condition, measured from their equilibrium positions, are indicated as $x_1(t), x_2(t)$ and assume positive values when the movement is towards right, otherwise they have negative values.
The equation of motion are:

$$m_1\ddot{x}_1(t) = (-k_1 - k_2)\, x_1(t) + k_2 x_2(t)$$

$$m_2\ddot{x}_2(t) = k_2 x_1(t) + (-k_1 - k_2)\, x_2(t).$$

The system can be written as:

$$M\ddot{\boldsymbol{x}}(t) = K\boldsymbol{x}(t) \tag{5.1}$$

where $\boldsymbol{x}$ is the displacement, M the mass matrix and K is the stiffness matrix:

$$\boldsymbol{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \qquad M = \begin{pmatrix} m_1 & 0 \\ 0 & m_2 \end{pmatrix} \qquad K = \begin{pmatrix} -k_1 - k_2 & k_2 \\ k_2 & -k_1 - k_2 \end{pmatrix}$$

Therefore, it is a constant-coefficient homogeneous system of second-order ODEs, whose analytical solution is:

$$\boldsymbol{x}(t) = \sum_{i=1}^{2} [a_i \cos(\omega_i t) + b_i \sin(\omega_i t)]\boldsymbol{v}_i$$

where $\omega_i = \sqrt{-\lambda_i}$ and λ_i are the eigenvalues of the matrix $M^{-1}K$; $\boldsymbol{v}_i$ are their eigenvectors. The constants a_i and b_i are determined by the initial conditions $\boldsymbol{x}(t=0) = \boldsymbol{x}_0$ and $\dot{\boldsymbol{x}}(t=0) = \dot{\boldsymbol{x}}_0$. Its minimum is $f(\boldsymbol{x}^*) = -1$ at $\boldsymbol{x}^* = (1.000, 3.946, 4.000, 3.286)$. However, it is possible to convert the Equation (5.1) into a system of first-order ODEs that can be solved using the fourth-order accurate Runge-Kutta time-marching method and varying the time-step Δt between the two fidelity levels. It is adopted a $\Delta t = 0.01$ for the high-fidelity model and a $\Delta t = 0.6$ for the low-fidelity model. The initial condition are set to: $\boldsymbol{x}_0 = (1\ 0)^T$ and $\dot{\boldsymbol{x}}_0 = (0\ 0)^T$. Masses and springs range within the interval [1,4].

Function	**Dimension**	**Budget**	**HF cost**	**LF cost**	**HF samples**	**LF samples**
Spring-Mass	4	400	1	1/60	4	10

Table 5.9: Spring-Mass experiment setup

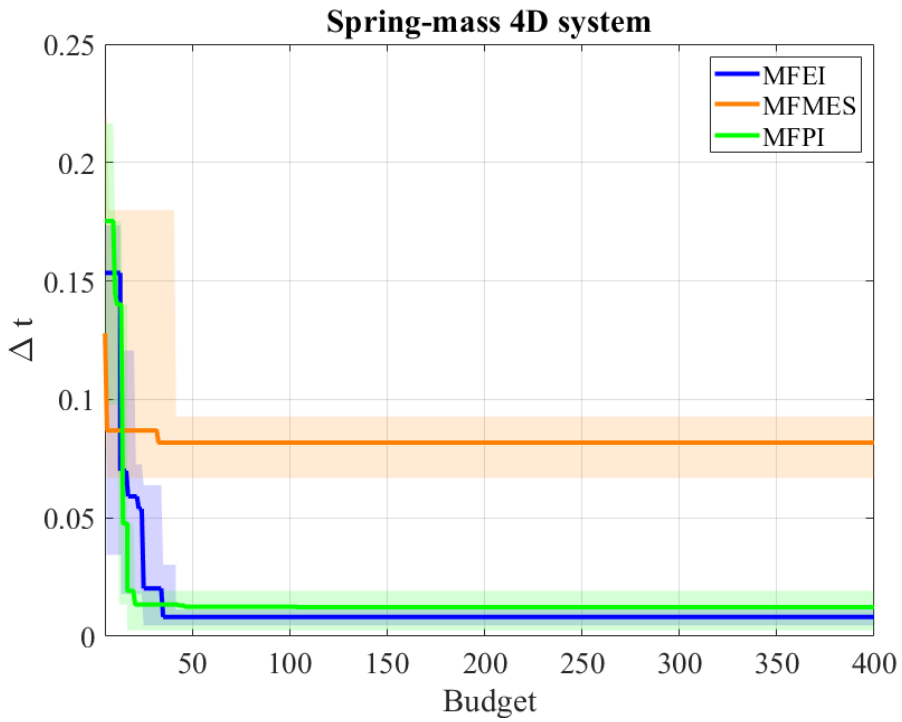


Figure 5.42: Comparison of MFEI, MFMES and MFPI on Mass-Spring system function.

5.9.1 Results

As previously explained for the Rosenbrock 5D function, also in this case only five tests have been run due to narrow resources. Moreover, the following figures will show only the seeding behaviour till the optimum found by the algorithms, because no further interesting information can be drawn from the observation of all the iterations till the end of budget. Comparing all the methods, it is possible to notice that the MFMES starts with a better sampling than the others, but then it cannot lead to reduce the residual error beyond 0.082. Whereas, MFPI, that starts with a grater residual error, is able to achieve a reduction till the value of 0.012, while the MFEI can further decrease until 0.008.
The MFEI widely employs the low-fidelity model, thus preferring an exploration approach, therefore 530 LF points are used with respect to the 150 HF ones. Just from the 80-th run a greater amount of high-fidelity patterns is used. A total of 136 iterations results useful to provide information about the area in which the optimum is placed. However, not all the tests find the optimum, indeed in Figure 5.42, there is not an annulment of the error.
The MFPI uses a greater number of total points to discover the domain, than the MFEI. It prefers to adopt the low-fidelity model, but a good trade-off between exploration and exploitation can be found all over the space of design. A cluster of points is visible at the extremes. Although the major number of samples, especially HF, and iterations, the multifidelity Probability of Improvement cannot reach better performances than the Expected Improvement, as seen in Figure 5.42.
The MFMES employs the narrowest set of points than the other methods, thus preferring to call the high-fidelity model. A wide range of samples is placed between the coordinates 2 and 3.5, while few points are spread near x^*. Indeed, this explains the worst performance of the MFMES method with respect to the others.

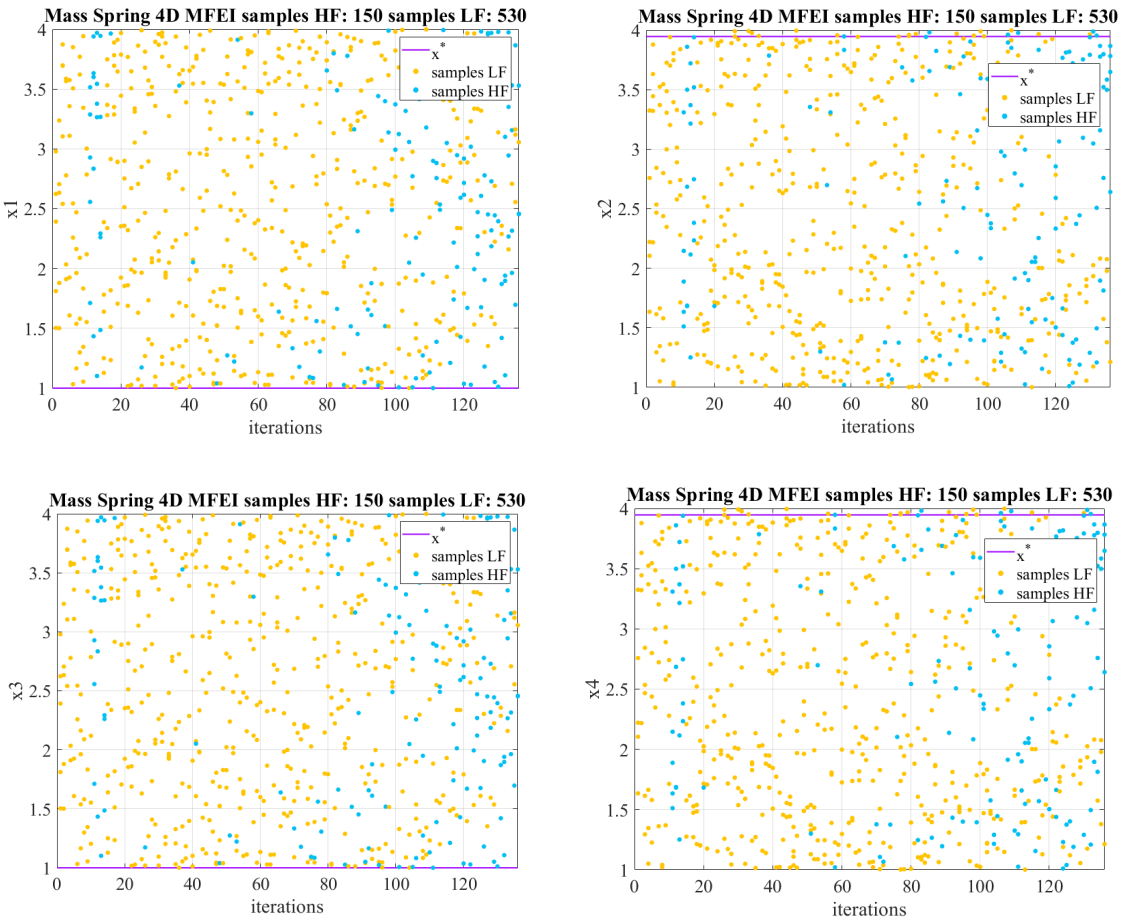


Figure 5.43: Distribution of MFEI samples over x.

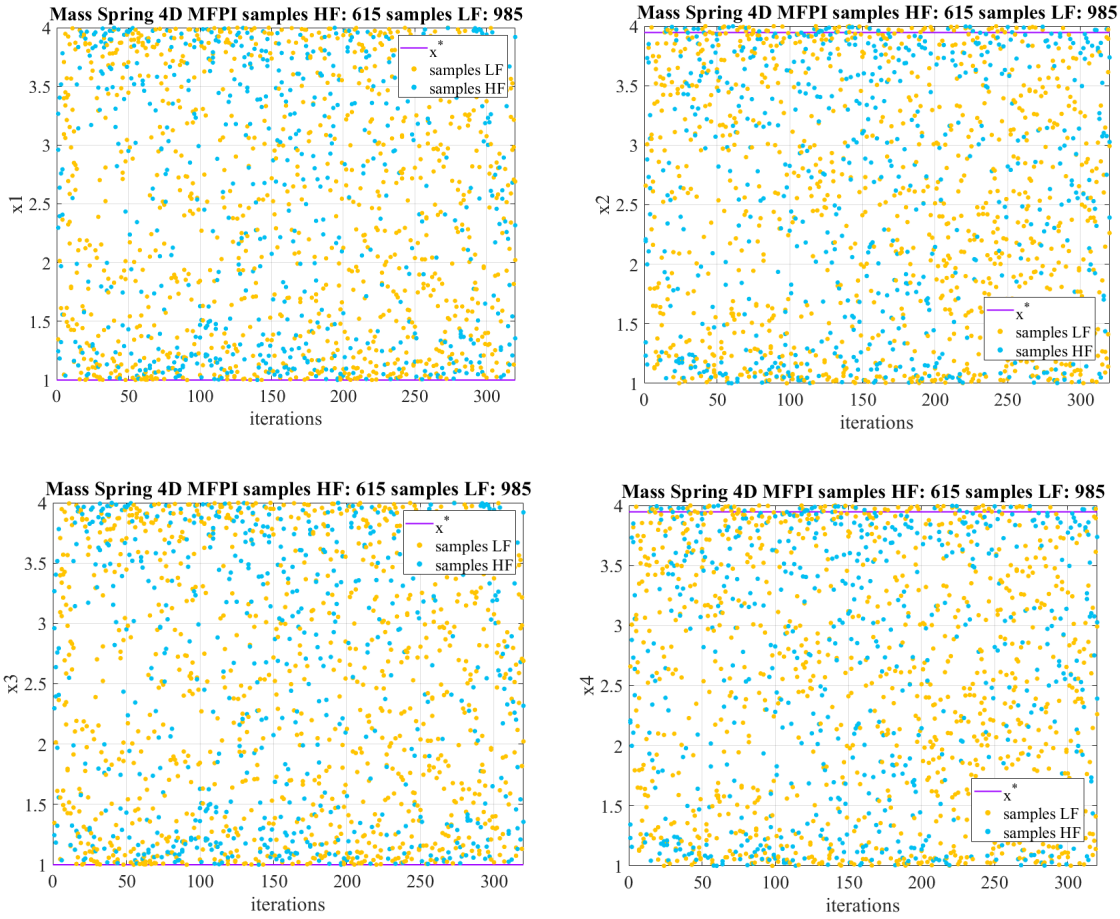


Figure 5.44: Distribution of MFPI samples over x.

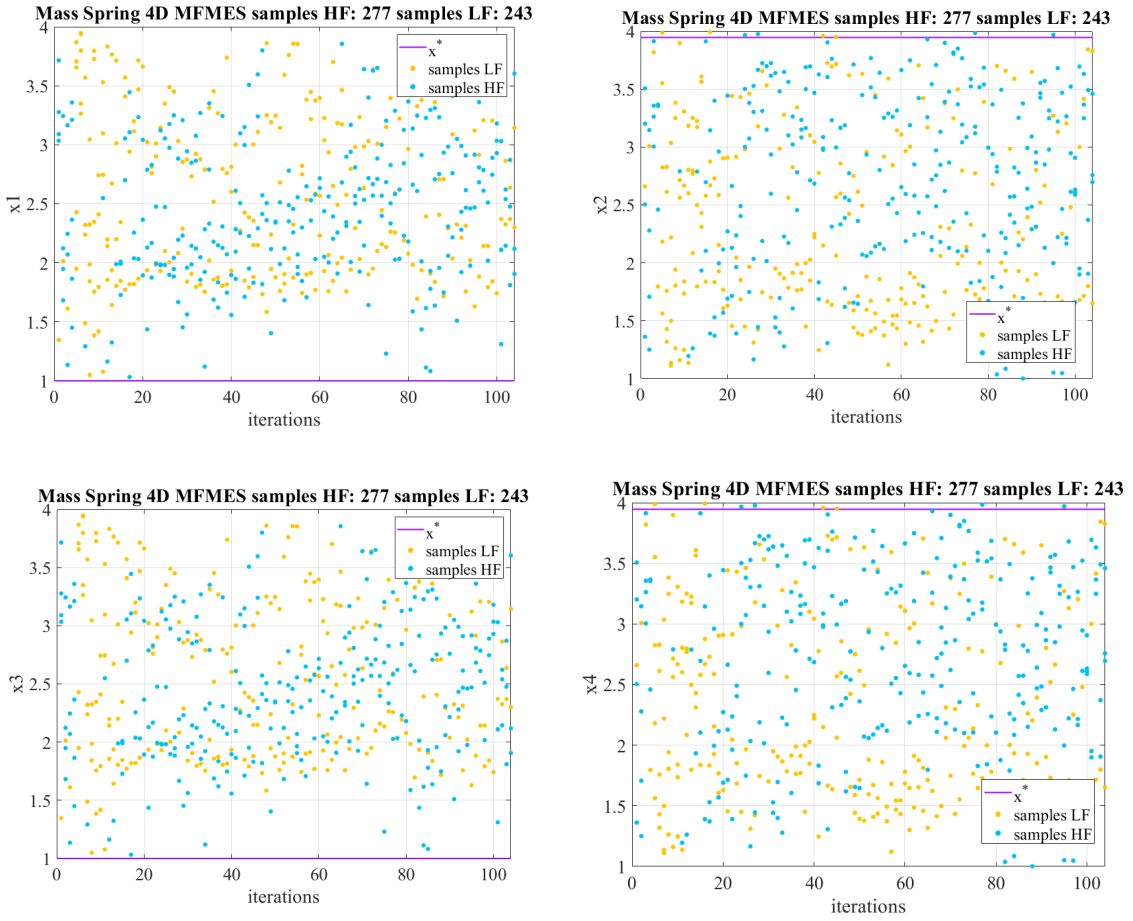


Figure 5.45: Distribution of MFMES samples over x.

5.10 Results overview

Comparing the three methods, the Multifidelity Expected Improvement results as the acquisition function with the best performance of all, because it generally requires the narrowest number of iterations and budget. Generally, in terms of minimization of the error Δ_t, both MFEI and MFPI functions have comparable behaviours, and tend to work efficiently to reduce it. Even if the Probability of Improvement may have a faster start, thus tending to converge more rapidly as seen for the Forrester, the Rastrigin and the Clark and Bae 3D functions, then the residual error reached may be slightly greater than that achieved by the Expected Improvement. At the same time, it is possible to obtain a lower residue by using the Probability of Improvement function on Clark and Bae 1D and Sinusoidal function, even if it took a worse initial dataset than the Expected Improvement. Also when applied to the Clark and Bae 2D function, the MFPI shows an improvement with respect to the MFEI, thanks to the higher amount of HF calls. This is an inherent characteristic of the MFPI formulation that uses a penalty to reduce the PI function when calling the low-fidelity model. This behaviour results useful for low-dimensional problems, because makes the surrogate more accurate via a stronger exploitation. Whereas, the exploration technique is preferred by the MFEI, even after finding the optimum location. This is advantageous also for higher dimensional problems (e.g. Rosenbrock 5D function), because thanks to the exploration with the LF model, the method can reduce the initial set of HF samples required. The Multifidelity Max-value Entropy Search, instead, needs always more budget, iterations and points to spread for converging. This is due to the Monte Carlo simulations employed to compute the expectation over the minimum, that add another stochastic ingredient to the optimization process. Hence, by comparing the three methods in terms of Δ_t and budget, the confidence interval of the MES is always the larger of all and reduces more slowly. Generally, it tries to decrease as possible the number of total calls of the models, so it is more open to discover the space of design with the high-fidelity model. However, for the Forrester, the Rastrigin and the Clark and Bae 3D functions, there is not a significant difference between high- and low- fidelity samples used, thus demonstrating a trade-off between exploration and exploitation. The method shows better performance than the MFPI when applied to the Rosenbrock 2D function. In this case, after the convergence, both the MFEI and MFMES demonstrate a repetition of sampling paths that can suggest possible improvements in the stopping criterion to use. It is important to highlight that the number of Monte Carlo simulations to use for MFMES has been set to 10, taking into account the settings employed in [83], also because no better performance have been observed by increasing this number with the dimensionality of the problems. Anyway, Takeno et al. used a greater set of initial points that can advantage the computation, so further tests may be useful to clearly understand this correlation, in order to notice if a bad setting has influenced the performance of the method.
For what concerns the problems with higher dimensionality, such as the Rosenbrock 5D and the Spring-mass System 4D, the lowest performance arise: a narrow number of test is able to reach the optimum, but all of them moves towards it. This may be due to the loss of accuracy of the surrogate, with increasing dimensionality, as well as the narrower number of statistical tests run, thus probably providing unfavourable sets of initial points.

Chapter 6

Application to L2 problem

In this chapter, the multifidelity methods are applied to a L2 problem. This is a category of intermediate-complexity problems, which are more realistic and detailed in representing real engineering problems than L1 problems. Therefore, their computational costs results greater than L1 problems.
Hereafter, an aerodynamic problem is depicted. The goal is to globally optimize the shape of a wing profile, that is managed parametrically, in order to reduce the drag rise and improve the stability of the airfoil when it approaches transonic regimes. The software adopted is developed by NATO-AVT-252 [89]. It is an open source code, written in *bash* and *awk* languages, and is made for UNIX operating systems. For this reason, it is to report that, at the current state of the art, several portability problems for Microsoft Windows operating system have been arising, thus the final choice to use the original configuration. The software package can also run on a Linux based virtual machine but, we install it by using a dual boot due to incompatibilities found about the communication between the virtual machine and the hardware to use. For what concerns the entire optimization process, it has been run on the academic cluster (HPC) that is Linux based.
The software takes in input the baseline airfoil shape, the design variables, the flow working conditions and the settings for a parallel Reynolds-Averaged Navier–Stokes (RANS) computation. Eventually, it can be set up also for handle Uncertainty Quantification shape parameters, but in this thesis this option is not adopted.
Then, the shape handler *WG2AER* uses these inputs to realize the new airfoil design (here, only single-component airfoils are implemented). *WG2AER* accepts in input a set of design variables that are:

1. the starting airfoil shape, $(x_0(s), y_0(s))$, which by default is a NACA 2412, but another airfoil can also be chosen by the users;

2. the modification functions, $y_i(s)$ selected among a set of 32 function types that include 6 polynomials, 12 Hicks-Henne, 2 Wagner and 12 user defined functions given by points;

3. the weights w_i to assign at each modification function.

Therefore, *WG2AER* code parametrizes the airfoil using a linear combination of an initial geometry and the shape modification functions chosen $y_i(s)$. In the case in which it is necessary to describe also geometry uncertainties, they can be added to

the formulation as Hicks-Henne bumps functions, $z_i(s)$. So, the formulation is:

$$
\begin{aligned}
y(s) &= k\left(y_0(s) + \sum_{i=1}^{n} w_i y_i(s)\right) + \sum_{j=1}^{m} U_j z_j(s) \\
x(s) &= x_0(s) \\
z_j(s) &= \sin^3\left(\pi s^{\frac{\log 0.5}{\log s_{b_j}}}\right)
\end{aligned}
$$

where k is the scale factor that controls the shape modification functions, and U_j are random variables that control Hicks-Henne bumps, and describe the uncertainty on shape and thickness of the airfoil.
Therefore, the *run_solver.sh* script, which wraps *WG2AER*, uses this formulation to build a parametric airfoil, whose aerodynamic performances are, then, analysed by means of the solver selected. It can be SU2, which is matched with the mesh generator GMSH, or XFOIL. SU2 is a free open-source software, written in C++, whose initial features have been developed by the Department of Aeronautics and Astronautics of Stanford University, but then, it has been updated with further capabilities also from all the community. It can solve optimization tasks and partial differential equations (PDE) and PDE-constrained optimization problems [97, 98]. This skill can be extended to solve multiphysics analysis and design problems and to realize a RANS solver which simulates the compressible, turbulent flows with which often aerospace engineering has to deal. Furthermore, SU2 is a suite of different modules, each of them can be used separately or combined. Hereafter, SU2 CFD module is adopted with the version 6.2.0 and has been coupled with the open source mesh generator GMSH. In context of optimization, it is necessary to generate a mesh with a peculiar grid for each geometry evaluated. For this purpose, an automatic procedure that calls GMSH at every new geometry to evaluate, is implemented and coupled to SU2. This is built taking into account a trade-off among calculation accuracy, fast CFD computation and the robustness of the procedure to minimize possible errors that may arise during the grid generation. The automated mesh generator takes in input the airfoil coordinates, some flow data as the dimensions of the fluid domain, the Reynolds number and a reference length, and the grid control parameters. It then returns a grid suitable for SU2, composed of triangles and quadrilaterals [89]. The parametric mesh generation is able to manage cell sizes and to refine accurately the wake, the corner points and the boundary layer by clustering grid points in these regions characterised by high gradients.
XFOIL, instead, implements a second order panel method, which depicts the airfoil surface with panels consisting of sources or vortex, coupled to a boundary layer integral module. The transition point from laminar to turbulent is determined using the e^N formulation [99]. The selection between the two solvers, is made on the basis of the working conditions chosen. The problem of interest is set in a transonic regime, so XFOIL has been excluded: it loses performances for $M > 0.5$.

6.1 Physics of the problem

6.1.1 Governing equations

The governing equations of a viscous fluid flow are the Navier-Stokes. Viscous flows are characterised by dissipative transport phenomena of friction, thermal conduction and mass diffusion, and so, by significant gradients of temperature, velocity and chemical composition. Navier-Stokes equations include the equation of the conservation of the mass, the momentum equation and the energy equation. For a compressible flow (i.e. a flow characterised by density variations not negligible) they are, respectively [100]:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \boldsymbol{V}) = 0$$

$$\frac{\partial (\rho \boldsymbol{V})}{\partial t} + \rho(\boldsymbol{V} \cdot \nabla \boldsymbol{V}) = -\nabla p + \nabla \cdot \boldsymbol{\tau} + \rho \boldsymbol{f}$$

$$\frac{\partial}{\partial t}\left[\rho\left(e + \frac{V^2}{2}\right)\right] + \nabla \cdot \left[\rho\left(e + \frac{V^2}{2}\right)\boldsymbol{V}\right] = \rho \dot{\boldsymbol{q}} + \nabla \cdot (k \nabla T) - \nabla \cdot (p \boldsymbol{V}) + \\ + \nabla \cdot (\boldsymbol{\tau} \cdot \boldsymbol{V}) + \rho \boldsymbol{f} \cdot \boldsymbol{V}$$

where: ρ is the density; $\boldsymbol{V}$ is the vector of velocity; e is the internal energy per unit mass, so $e + \frac{V^2}{2}$ is the total energy; k is the thermal conductivity and is related to the Fourier's law (which relates the heat flux, due to thermal conduction, to local temperature gradient); $\dot{q}$ is the heat flux (the heat transferred by conduction into the fluid); $\boldsymbol{\tau}$ is the stress tensor, which contains shear and normal stresses both related to velocity gradients; $\rho \boldsymbol{f}$ and $\rho \boldsymbol{f} \cdot \boldsymbol{V}$ are the body forces (e.g. weight due to gravitational force) and the rate of work they do on the fluid element.
These equations are a strongly coupled system of nonlinear PDEs: it results very difficult to solve analytically, indeed, a close-form solution has not been found. The reachable accuracy, with Navier-Stokes, depends on the prediction accuracy adopted for describing the main characteristics of a turbulent flow. For this purpose, the RANS equations are the most commonly used. Therefore, they derive from a time averaging of the Navier-Stokes equations and approximate the exact equations. They are obtained starting from the Reynolds decomposition of the flow variables into mean and fluctuating parts, and then inserting these decomposed variables into the Navier-Stokes equations and averaging the equations [101]. This produces the Reynolds-stress tensor $\boldsymbol{\tau}_{i,j} = -\overline{\rho u_i' u_j'}$ (for newtonian fluids, shear stresses are considered proportional to the velocity gradients [100]).
However, employing the RANS also means to find a way to close these equations, by computing the components of the unknown Reynolds stress tensor: this is the task of a turbulence model [102]. Hence, this computation is not trivial to handle numerically and may require very high computational costs to solve the transport equations. The turbulence model allows to assemble a simplified way to compute the transport equations of the Reynolds-stress tensor. The choice of the turbulence model is important because affects the computational time required for the aerodynamic simulation. Following the problem settings implemented in [89], here a Spalart-Allmaras model [103] is adopted. It is a one-equation linear model. The term 'one-equation' indicates that only one turbulent scale, or a combination of

them, is evaluated by means of the solution of a transport equation. Therefore, only an additional eddy viscosity transport equation (a PDE) is added to be computed in conjunction with the RANS.
The model provides an empirical equation that describes the evolution of the turbulent kinematic viscosity, $\nu_t = \mu_t/\rho$, in order to model production, transport, diffusion and destruction of ν_t. According to Boussinesq hypothesis, the turbulence can be described as an increased viscosity. The total viscosity can be divided into a laminar, μ_{dyn}, and a turbulent one μ_t, but while the first is evaluated by using the Sutherland's law, the latter μ_t, is computed following the turbulence model chosen [104]. For Spalart-Allamaras, it is:

$$\mu_t = \rho\tilde{\nu} f_{\nu 1}, \quad f_{\nu 1} = \frac{\chi^3}{\chi^3 + c_{\nu 1}^3}, \quad \chi = \frac{\tilde{\nu}}{\nu}, \quad \nu = \frac{\mu_{dyn}}{\rho}$$

where $c_{\nu 1}$ is the heat capacity at constant volume. The additional transport equation, can be expressed as:

$$\frac{\partial \rho\tilde{\nu}}{\partial \rho} + \frac{\partial}{\partial \boldsymbol{x}} \cdot (\rho\tilde{\nu}\boldsymbol{u}) = M(\tilde{\nu})\tilde{\nu} + P(\tilde{\nu})\tilde{\nu} - D(\tilde{\nu})\tilde{\nu}$$

where $M(\tilde{\nu})\tilde{\nu}, P(\tilde{\nu})\tilde{\nu}$ and $D(\tilde{\nu})\tilde{\nu}$ are respectively the diffusion term, the production source term and the wall source destruction term by which the $\tilde{\nu}$ is modeled, thus providing the eddy viscosity μ_t evaluation that has to be inserted in equation (6.1), in order to solve the governing equations [105].

$$\boldsymbol{\tau}_{i,j} = (\mu_{dyn} + \mu_t)\left(\left[\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{x}}\right] + \left[\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{x}}\right]^T - \frac{2}{3}\left[\frac{\partial}{\partial \boldsymbol{x}} \cdot \boldsymbol{u}\right]\boldsymbol{I}\right) \tag{6.1}$$

The same procedure is to apply to the other components of velocity, in order to describe all the tensor $\boldsymbol{\tau}$.

6.1.2 Transonic regime and supercritical airfoils

In the preliminary phase of design, it is important to analyse the pressure field around an airfoil, because it aims to evaluate aerodynamic performances, structural loads and stability problems with which a full vehicle design needs to deal.
Assessing the pressure distribution over the surface is essential to compute the corresponding force. It is divisible in two components that are parallel and perpendicular to the flow stream (V_∞): they are, respectively, drag D, and lift, L. The resultant force generate also a moment, the pitching moment M, with respect to a point located along the chord. It assumes positive values when tends to increase the angle of attack α. Therefore:

$$C_p = \frac{p - p_\infty}{\frac{1}{2}\rho_\infty V_\infty^2}$$

$$C_l = \frac{L}{\frac{1}{2}\rho_\infty V_\infty^2 l}$$

$$C_d = \frac{D}{\frac{1}{2}\rho_\infty V_\infty^2 l}$$

$$C_m = \frac{M}{\frac{1}{2}\rho_\infty V_\infty^2 l^2}$$

where C_p stands for the pressure coefficient, p and p_∞ are, respectively, the pressure evaluated at a generic point of the surface, and the freestream static pressure. At the denominator, there is the dynamic pressure, that is function of the density ρ and of the velocity V of the undisturbed stream. For the lift, drag and pitching moment coefficients, the denominator present also a reference surface S of the body of interest (also multiplied for the arm of the force M, in the case of C_m). In a 2D context like that of an airfoil, it can be expressed as $S = l \times 1$, where l is a reference length, generally the chord c for an airfoil. Integrating the C_p, allows to evaluate lift and drag forces, indeed, the area marked by the pressure coefficient, represents the resultant force coefficient.

The pressure distribution is characterised by pressures lower than that of the

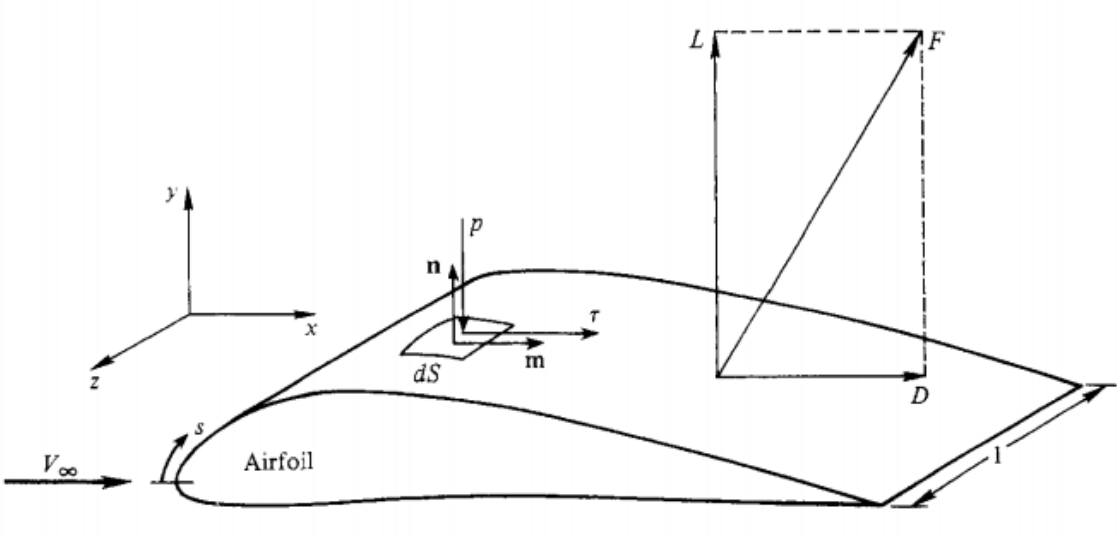


Figure 6.1: Aerodynamic resultant force and its resolution into lift and drag [106].

freestream, on the upper surface of the airfoil, and higher pressures on the lower surface. This is produced by the greater curvature of the upper surface, that accelerate the flow around the leading edge, leading to a reduction of the static pressure (i.e. expansion waves). Then, moving from the point of maximum thickness to the trailing edge, increasing pressures arise to match with the pressure at the trailing edge. For these reasons, the flow on the top side of the airfoil runs faster than that on the bottom side. For what concerns the lower surface, a stagnation point is here located, and represents the point in which the pressure is the highest of all the distribution. The local velocity here is zero, because all the kinetic energy is converted into static pressure isentropically. Moving about the trailing edge, a reduction of the pressure can be observed. Hence, a pressure coefficient equal to zero, represents the condition in which the pressure over the airfoil matches the undisturbed flow pressure. Since the suction force on the upper surface is greater than the pushing force on lower one, the first contributes more on the lift generation.

The pressure field is also function of α, indeed, if larger values of the angle of attack are used, a strong reduction of pressure can be seen on the upper surface, nearby the leading edge, whereas a strong increase can be found on the lower surface. This produces an increase of the lift, but, for high α, can lead to the boundary layer separation, and so, the stall (a rapid drop of lift).

According to Anderson [106], a transonic regime represents a flow condition in

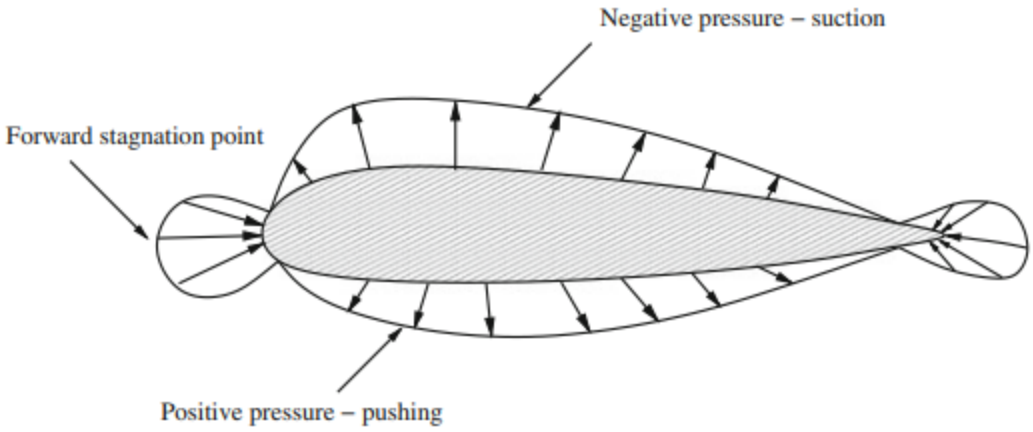


Figure 6.2: Pressure distribution over an airfoil with $\alpha = 0°$ [107].

which the Mach of the freestream flow is subsonic, but really near to the unity (generally $0.8 \leq M_\infty \leq 1.2$), thus allowing that a flow expansion, over the upper surface of the airfoil, can make the flow locally supersonic. It is a regime characterised by a

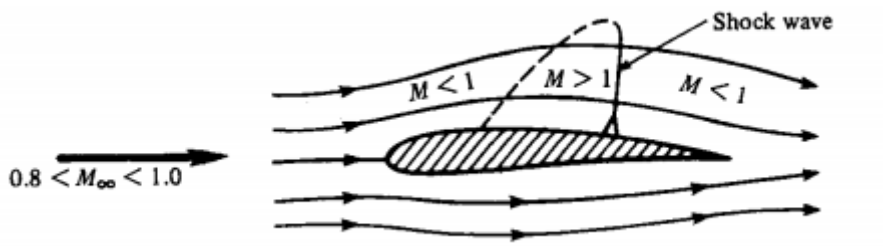


Figure 6.3: Transonic regime [106].

mix of subsonic and supersonic zones. Figure 6.3 summarizes the main features of a transonic regime, in which it is possible to observe the Mach reduction to subsonic conditions, after the generation of the shock wave.
As said, towards the upper surface of the leading edge, there is the acceleration of the flow by means of an infinite series of expansions. When sonic Mach is achieved locally (i.e. it means that the freestream exceeds the critical Mach number), the expansion waves tend to reflect on the sonic line (Figure 6.4), moving back towards the surface as compression waves (i.e. with increased temperature, pressure and density), thus decelerating the flow. Then, they are reflected again by the surface, as still more compression waves [108]. The presence of waves with opposite behaviours (deceleration and acceleration), can induce the deceleration of the flow that, up to this point, have been accelerating. This yields a gradual packing of perturbations, leading to the production of a shock wave. Across it, discontinuities and irreversible changes in flow properties are generated. It is a homoentalpic phenomenon, i.e. it does not induce a variation of the total temperature between the upstream and the downstream of the wave. However, the total pressure downstream reduces, because of the dissipative nature of the wave (presence of viscous effects and heat fluxes). At the same time, a shock wave produces high and sudden gradients, which increase all the static physical quantities (e.g. pressure, density, temperature) downstream.

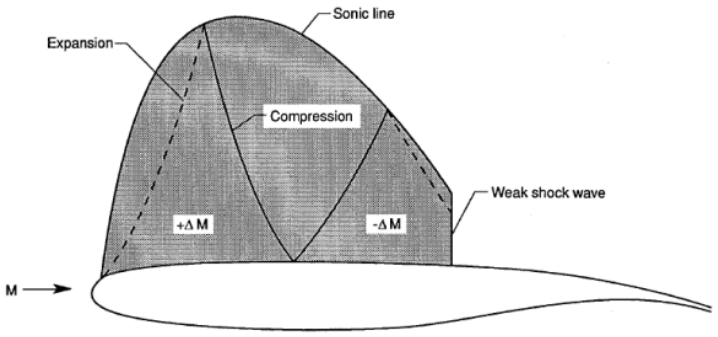


Figure 6.4: Transonic flow over supercritical airfoil [109].

Adverse gradients of pressure ($dp/dx > 0$) arise after the shock wave to match, more or less gradually, the pressures at the trailing edge (*Kutta-Joukowski condition*). In transonic conditions, trying to balance the influence of expansion and compression waves is essential to minimize the drag rise after a shock wave, without losing performance in subsonic regime. Managing appropriately the geometry of the profile is the solution to improve the location of the shock wave, thus reducing its strength. This is the reason why supercritical airfoils are used. Their advantages lie in the ability to improve the aerodynamic efficiency also for Mach greater than the critical Mach number. Figure 6.5 shows the greater weakness (i.e. reduced leap of pressure) of a shock wave over a supercritical airfoil with respect to a conventional one, because of the reduced curvature of the upper surface. Hence, weakening a shock wave can limit the interaction between the increased pressure, due to the shock wave, and the adverse gradient pressure of the trailing edge. It is important to limit this phenomenon, because it could lead to the boundary layer separation, providing an increase of drag, buffeting and stability problems [108]. Studying this flow regime is of practical interest for both military and commercial aircrafts, because it is important to handle the instability produced by the rapid increase of drag at transonic speeds. In this flight regime, the better performances of a supercritical airfoil, with respect to a conventional one, arise from its rounder leading edge, its reduced curvature, the maximum thickness retreat towards the aft, and its substantial aft camber. In this way, it can increase the drag divergence Mach number and the critical Mach number, move more aft the location of the shock wave, while keeping acceptable value of maximum lift at low speed and predictable stall characteristics [108]. According to Harris [108], the large leading edge can produce stronger expansions, in order to be reflected back as compression waves to limit the flow acceleration both over the upper and lower surface. At the same time, this allows to minimize the surface curvature. A flatter upper surface leads to a reduction of the flow acceleration over it, thus maintaining a quite constant supersonic flow (a pressure plateau). In this way, there is the generation of weaker shock waves, because the compression waves have to overcome a narrower amount of expansion waves. Thanks also to the maximum thickness retreat, the shock is delayed more afterward, thus reducing the airfoil section in which a separation could arise. However, this region of low curvature cannot be extended excessively because, towards the aft, it could create a great trailing edge slope, thus increasing an adverse pressure gradient. The lift lost through the flatter upper surface, is recovered by means of a greater curvature on

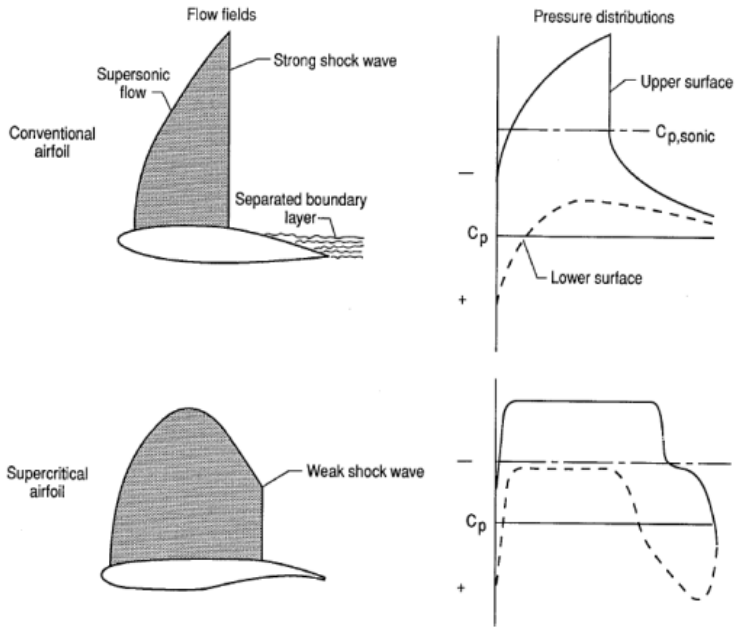


Figure 6.5: Comparison between conventional and supercritical airfoils in transonic regime [109].

the rear portion of the airfoil [110]. A thick trailing edge is used to retard, or avoid, flow separation by reducing the difference between the pressure coefficients of upper and lower surface, near the aft. Indeed, the cusp at the aft lower surface leads to a rapid increase of pressure that, then, becomes more smooth. This design is selected following the Stratford criteria [111], and allows to increase lift, while energizing the flow to impede a boundary layer separation. It can increase the circulation over the airfoil, thus providing a smaller incidence to satisfy lift constraint. The trailing edge presents quite the same slope, i.e. a small angle, on both upper and lower surface, in order to minimize the pressure recovery of its upper surface.

6.2 Problem settings

After describing the flow conditions in which the L2 problem is set up, and the improved performances of supercritical airfoils in these regimes, it is necessary to tune conveniently the software settings. This concerns: the formulation of the optimization problem, whose aim is to minimize the drag coefficient C_d subject to previously discussed constraints on the geometry; the selection of the modification functions to shape the profile, the choice of CFD solver, and so grid generator, the settings and the adoption of the best suitable setup for the multifidelity optimization process with respect to time and computational resources available.

The optimization problem is formulated as [89]:

$$\begin{aligned} \underset{w \in W \subseteq \mathbb{R}^n}{min} \quad & C_d \\ \text{subject to :} \quad & C_l = 0.824 \\ & -0.1 \leq C_m \leq -0.01 \\ & t/c = 0.1211 \\ & r \geq 0.007c \\ & \tau \geq 5^\circ \\ & \tau_{85}/c \geq 0.02 \end{aligned}$$

where: t/c is the maximum airfoil thickness fixed to 12.11% of the chord c; r is the radius of the leading edge, expressed as function of the chord; τ is the trailing edge angle and t_{85}/c is the thickness of the airfoil at 85% of the chord. However, the equivalent unconstrained formulation is implemented:

$$\begin{aligned} \underset{w \in W \subseteq \mathbb{R}^n}{min} \; C_d + k_1 p^+(C_m, -0.1) + k_1 p^-(C_m, -0.01) + k_2 p^+(r/c, 0.007) + \\ + k_3 p^+(\tau, 5) + k_4 p^+(\tau_{85}/c, 0.02) \end{aligned} \tag{6.2}$$

where, the quadratic penalty definition is used:

$$p^+(x, y) = \begin{cases} 0 & \text{if} \quad x \geq y \\ (x-y)^2 & \text{if} \quad x < y \end{cases}$$

and

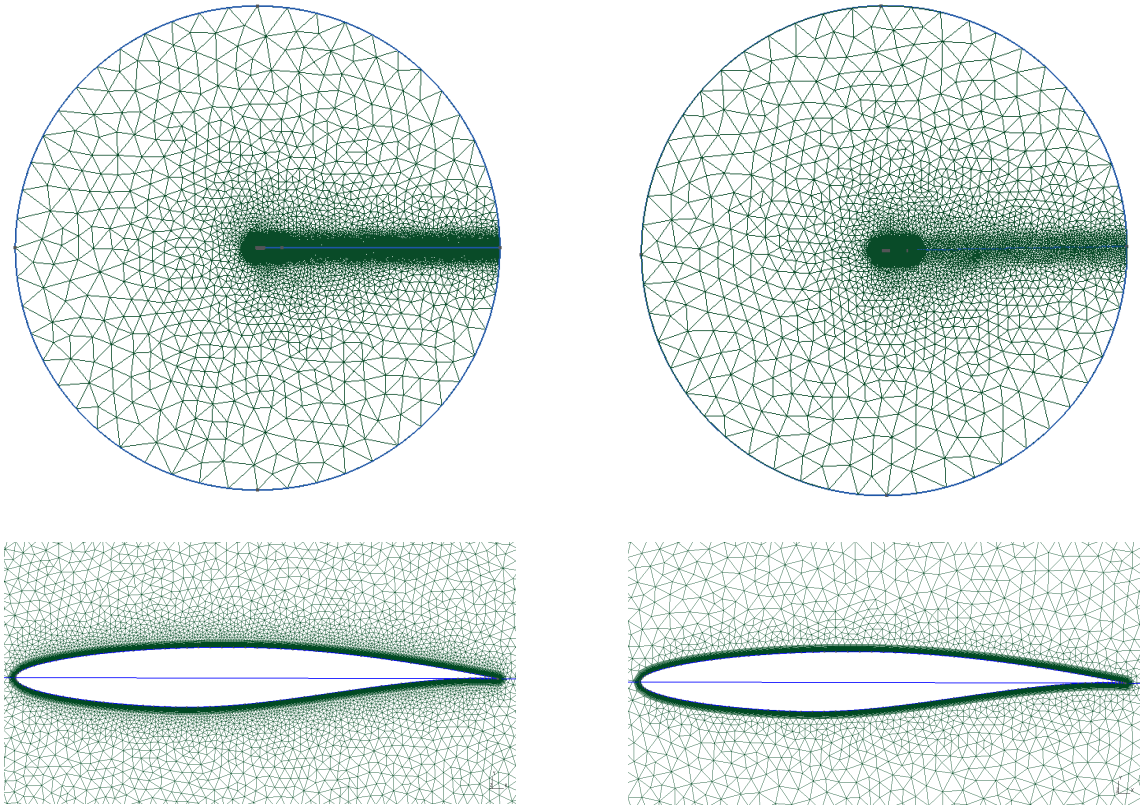
$$p^-(x, y) = \begin{cases} 0 & \text{if} \quad x \leq y \\ (x-y)^2 & \text{if} \quad x > y \end{cases}$$

with $k_1 = 1000, k_2 = 5000, k_3 = 10$ and $k_4 = 30$. This formulation is missing of the constraint on the thickness, because it is automatically satisfied scaling the airfoil after its parametric modification. The CFD simulations run with a fixed C_l, therefore, this constraint can be satisfied because the angle of attack, α, is set up as a free parameter. The bounds on the pitching moment are necessary because, to converge towards the prefixed C_l, the optimization process will tend to strongly change the camber of the trailing edge, so as to not affect meaningfully the drag. This causes a nose-down pitching moment, thus providing the possibility to increase the trim drag, a problem that can affect the stability in a full aircraft design context [112]. These constraints derive from the necessity to obtain a nearly supercritical airfoil because, even if it may result more complex to design by a structural point of view, it is the most suitable shape for transonic conditions. This explains the choice of the supercritical airfoil, RAE 2822, as baseline. Furthermore, it is an airfoil widely employed for validating turbulence models.

6.2.1 Low- and High-fidelity model setup

Before describing in depth the main settings of the shape optimization process, the features of the two different fidelity model are depicted.

The difference between the high- and low-fidelity models (HF and LF) lies in the grid coarseness. The grid density parameter is tuned through the input factor *ELEMENT_SCALE*: the larger this value is, the coarser the grid will be. It was first set to 2.5 for the high fidelity function, and to 12 for the low-fidelity but, due to limited time and resources, the coarseness of the high-fidelity grid has been then increased to 5.5. Comparing the pressure distribution over the airfoil by means of the Pres-



(a) ELEMENT_SCALE = 5.5: around 82000 elements

(b) ELEMENT_SCALE = 12: around 43000 elements

Figure 6.6: Comparison of the two different grids chosen for HF and LF

sure Coefficient (Figure 6.7), the difference between the two level of fidelity can be noticed. In particular, an increase in the suction force on the upper surface of the airfoil arises in the low-fidelity model, with a peak at the leading edge that is moved more forward. Due to this higher expansion over the top of the airfoil, it is showed a change of position of the shock wave: the low-fidelity model anticipates (along x/c) the perturbation, that is at about the 50% of the chord, while the high-fidelity model places it at roughly the 55-60% of the chord. A thickening of the boundary layer at the shock foot can be observed, due to the abrupt pressure rise, and larger wake is visible in the LF model. Indeed, the figures 6.7b and 6.7d show an increase of the eddy viscosity μ_t value in the LF. After the pressure leap, the low-fidelity model depicts a lesser recompression than the high-fidelity model, and a major pressure load on the upper surface of the aft. Whereas on the aft lower surface, a slightly lower pressure load is evaluated by the LF than the real one. A summary can be made by analysing the Mach distribution: LF model concentrates the Mach rise in a more forward region than the HF model, thus providing an anticipation of the thickening boundary layer. The bottom side of the airfoil sees an acceleration, but without

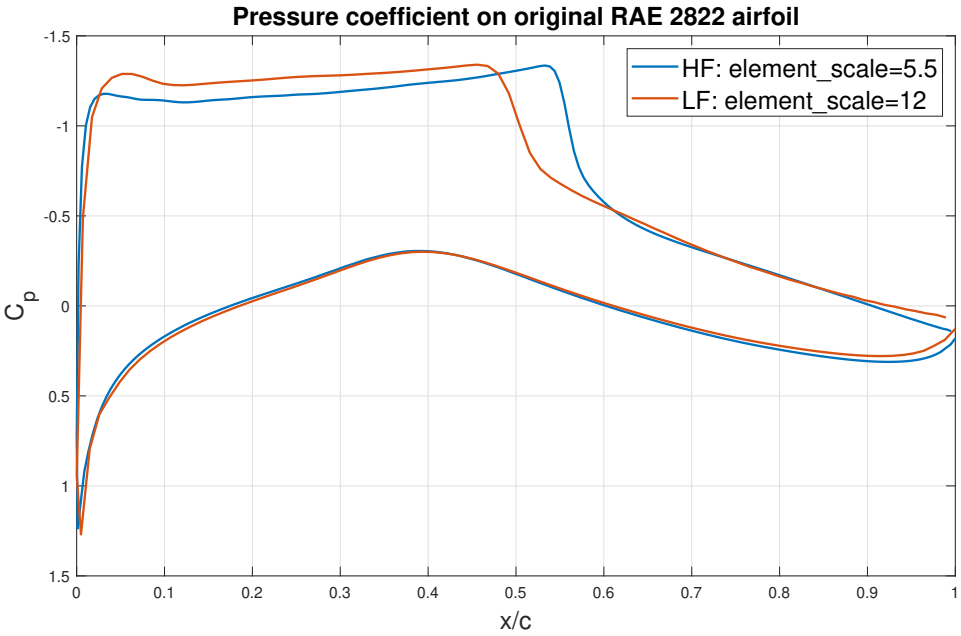
reaching sonic Mach, therefore no shock waves arise. From the CFD analysis of the original RAE 2822, it is possible to notice that the convergence towards the same C_l is obtained through an increase of the angle of attack, in the case of the low-fidelity model, therefore, a gain of C_d is visible and a corresponding loss of efficiency too.

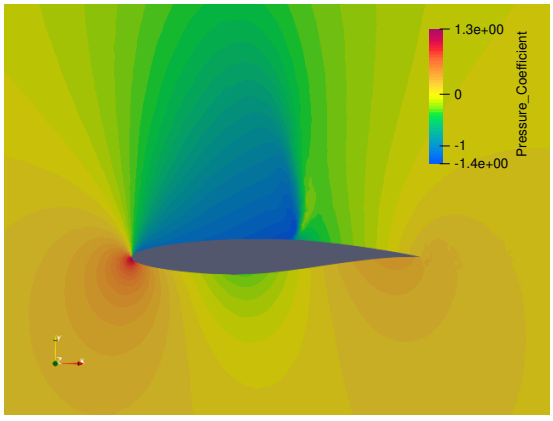
eval_obj.out	**HF model**	**LF model**
Mach	0.734	0.734
α	2.9121°	3.5244°
C_l	0.8239	0.8241
C_d	0.0226	0.0343
C_m	-0.0955	-0.0834
Efficiency, $E = C_l/C_d$	36.4742	24.0055
Max thickness, t/c	0.1211	0.1211
Leading edge radius, r	0.0079c	0.0079c
Trailing edge angle, τ	8.6808°	8.6808°
Thickness at 0.85c, τ_{85}/c	0.0318	0.0318

Table 6.1: Output file of the analysis on RAE 2822 with two different fidelity levels.

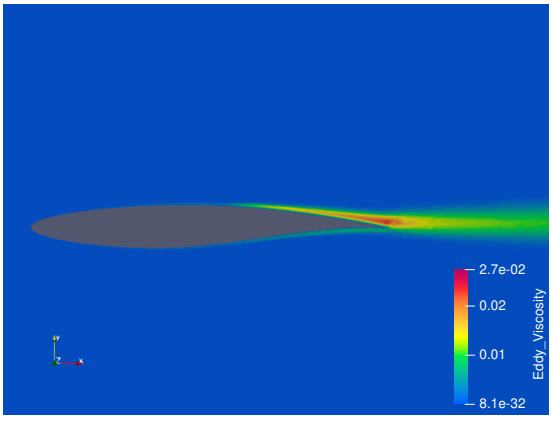
6.2.2 L2 software package setup

It is decided to implement a problem that concerns 20 design variables, that are ten modification functions (polynomials, Hicks-Henne and Wagner) for both upper and lower airfoil surface (see Figure 6.9). The related weights range within the interval $[-0.5, 0.5]^D$ and are picked by the Latin Hypercube of the optimization algorithm described in Section 4.2. This range has been found with a trial and error method. It was initially assumed to vary within the interval $[-1, 1]^D$, according to experts' opinions, but a great amount of non converging configurations were found with this particular combination of settings. For this reason, the necessity to narrow the interval of interest arises.
The software that manages the entire L2 package, takes in input a file (called *eval_obj.in*) that includes several parameters to tune: they concern the starting geometry and the weights of the modification functions, the working conditions, the GMSH and SU2 setup (i.e. its configuration file) and the number of CPUs and node size adopted for a parallel computation. The last are selected on the basis of the computational resources available on the academic computing service of Polytechnic of Turin, HPC.
For what concerns the flow conditions, in addition to the lift coefficient and the Mach number of the freestream, they are also expressed by the Reynolds number. Here, it is adopted a value of 6.5 millions, so it is a fully turbulent regime, whose typical features are non stationary and messy fluctuations that affect the wake. In this framework, inertial effects dominate on viscous effects, skin friction increases, and stall and separation are delayed and sufficiently inhibited. However, increasing the angle of attack may lead the separation point on the upper surface (nearby the

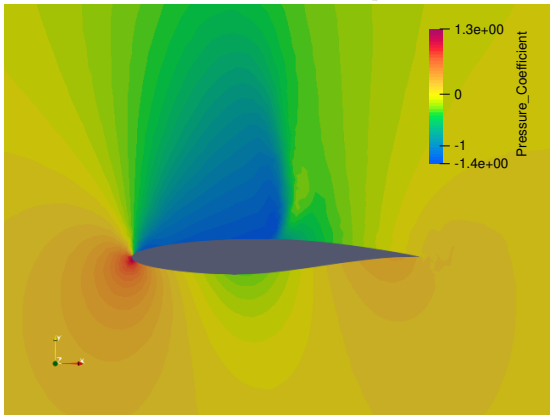




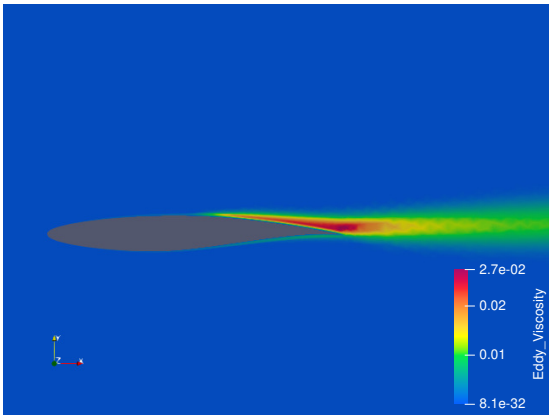
(a) HF model C_p



(b) HF model eddy viscosity μ_t

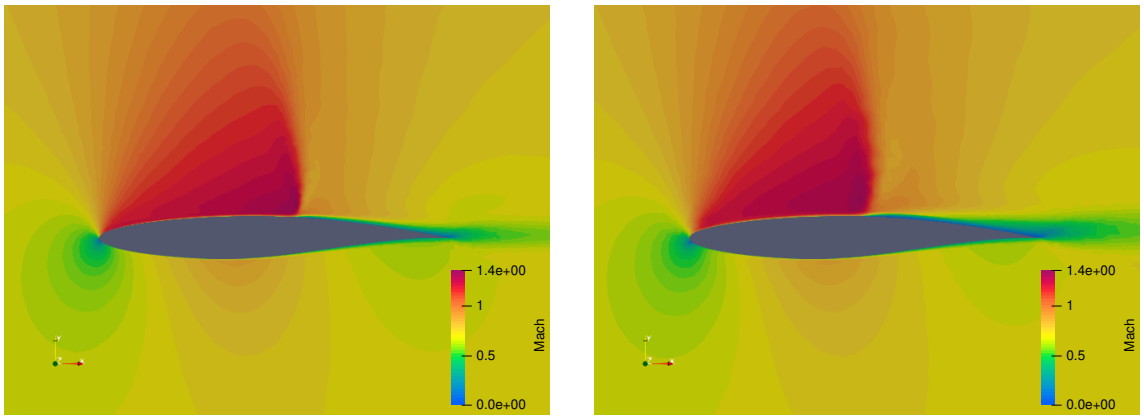


(c) LF model C_p



(d) LF model eddy viscosity μ_t

Figure 6.7: C_p and μ_t comparison, over the original RAE 2822, with HF and LF models.



(a) HF Mach number (b) LF Mach number

Figure 6.8: Mach number comparison, on the original RAE 2822, with HF and LF models.

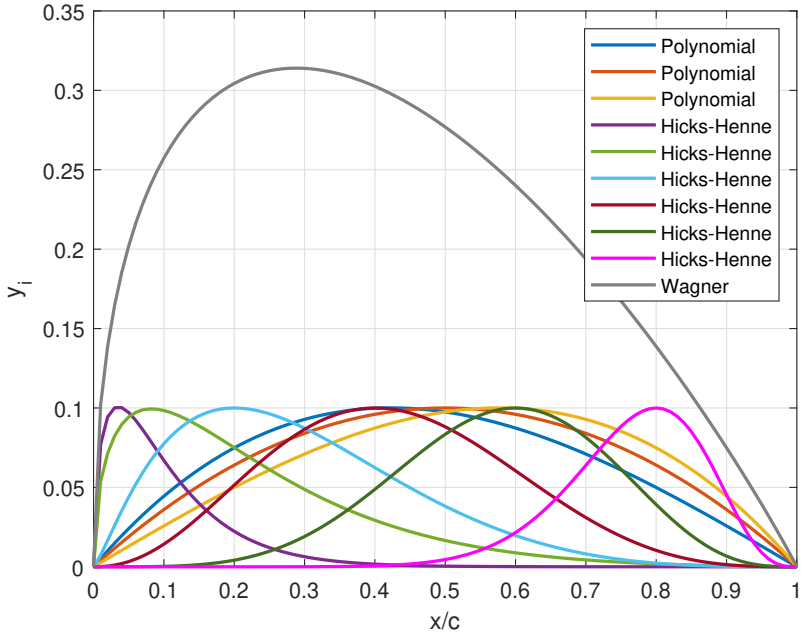


Figure 6.9: Modification functions adopted

trailing edge) to move towards the leading edge, thus bringing forward the separation and making the wake larger.
The grid generator settings include the parameter *Farfield* which sets the farfield boundary location (measured in chords) in order to delete edge effects. The farfield extension measures how much influence the boundary has on computation of forces (i.e. on the numerical solution). Farfield grid size is set by means of the parameter *Grid size*. Then, it is adopted by SU2 for setting the boundary conditions.
For what concerns the solver setup, it is necessary to choose: the maximum number of iterations needed to stop the simulation *MAX_CFD_ITER*; the convergence criterion, which is determined by the residual evaluated on the density *RES_RED_CFD*, so that the solver will stop when a residual reduction of three order of magnitude is found, with respect to the initial value; the CL drive definition, that is the se-

lection of a fixed lift mode, the corresponding number of times the angle of attack is updated within a fixed CL problem (i.e. between two angle of attack updates) *UPDATE_ALPHA* and the number of iterations to evaluate $dC_l/d\alpha$ at the end of the fixed CL simulation, *ITER_DCL_DALPHA*.
The flow is considered fully turbulent and the *Spalart-Allmaras* model is used, with a II order *Monotone Upstream-Centered Scheme* for Conservation Laws. It is a finite volume discretization method (FVM) that improves the accuracy of the numerical solutions till the second-order, in order to be able to capture also discontinuities and high gradients. The finite volume method (that aims to replace the partial derivatives contained in governing equations) is used to discretize in space the RANS. Furthermore, SU2 allows to decouple the integration in time and space of the governing equations, thus providing the possibility to choose two different integration methods. Hereafter FVM is used for spatial integration, whereas Euler implicit is chosen for time integration, by using Generalized Minimal Residual method (GMRES) with a minimum error set up to 10^{-6}, as a linear system solver.
An adaptive CFL number is set up, starting from an initial value of 2.5, where CFL stands for *Courant-Friedrichs-Levy* criterion. It can improve convergence, taking into account the time step adopted for the implicit/explicit numerical method (in this case, Euler Implicit) with respect to the distance between mesh elements. It establishes that the information acquired from a given mesh element, during the timestep length, must be limited only to immediately close cells. The output file, *eval_obj.out*, contains: possible errors arisen during the processing of GMSH or SU2; the Mach value at which the computation is run; the angle of attack that satisfies the constraint on C_l, the actual lift coefficient, its average over the last iterations (it should be nearly equal to C_l) and its standard deviation σ; the drag and pitching moment coefficients; the aerodynamic efficiency C_l/C_d; the residual reduction and the geometrical parameters adopted for the minimization problem, previously described (i.e. $r, t/c, \tau, \tau_{85}/c$).
To incorporate the performance of the software just described within the optimization process already applied to L1 problems in Chapter 5, two Matlab functions for the low- and high-fidelity models are implemented.
They write, on the input file, the initial design points (weights) picked from the domain, then they launch the script *run_su2.sh* and, when the aerodynamic simulation is completed, read the outputs from the related file *eval_obj.out* to compute the drag

Working conditions			
CL=0.824	Mach=0.734	Reynolds=6.5millions	T=216.65K
Shape definition			
Baseline=RAE 2822	Max thickness=0.1211	W(0),...,W(19)=provided by the optimization process	U(0),...,U(19)=0
Grid generator setup			
Lref.=1.0	ELEMENT_SCALE=5.5 HF ELEMENT_SCALE=12 LF	Grid size=5	Farfield=40
SU2 solver setup			
MAX_CFD_ITER=20000	RES_RED_CFD=3	UPDATE_ALPHA=20	ITER_DCL_DALPHA=500

Table 6.2: L2 problem settings

coefficient, following the minimization problem in Equation 6.2. This value is the m-th observation that is used into the optimization process (Section 4.2). However, a brief summary of this algorithm can be found in Figure 6.10, while the optimization setup is described in Table 6.3, remembering that 'MC' stands for the number of Monte Carlo simulations used for the Multifidelity Max-value Entropy Search.

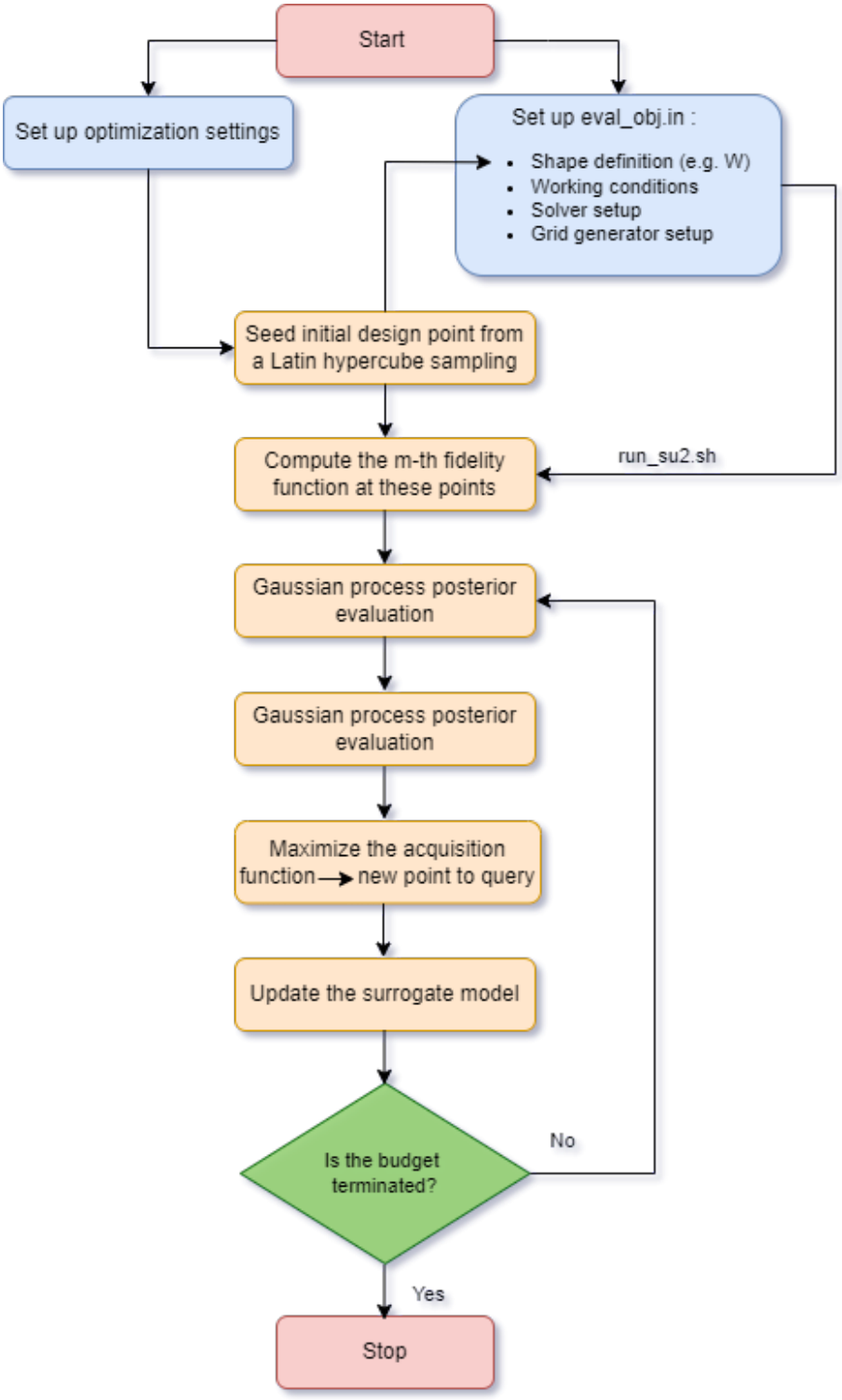


Figure 6.10: L2 problem integrated into the optimization process.

Function	Dimension	Budget	HF cost	LF cost	HF samples	LF samples	MC
L2	20	300	1	0.46	50	250	10

Table 6.3: L2 experiment setup

Following the guidelines of the international research group, the budget is to set up proportionally to the dimensionality of the problem but, due to narrow time and inadequate computational resources, it is preferred to reduce it to just 300, in order to save time for computing more tests with statistical purposes. The evaluation cost for the low-fidelity is obtained as a ratio between the grid coarseness of the high- and low-fidelity functions. The number of Monte Carlo simulations for the Multifidelity Max-value Entropy Search, follows the indication given by its developers.
The number of initial low- and high-fidelity samples has been found by means of a trial and error process: first, 10 HF and 250 LF samples were selected, but because of the lack of improvement of the Multifidelity Max-value Entropy Search method, the high-fidelity dataset has been increased to 50 samples.

6.3 Results

This section first shows the analysis run by using 10 initial HF samples and 250 LF ones starting from the same initial dataset, then demonstrates the results obtained by using the final setup concerning 50 initial HF samples and 250 LF.
All the methods, belonging to the same test case, share an equal set of initial points thus avoiding to facilitate a method more than the others.
In the case of 10 initial HF samples, this set results already advantageous with respect to the baseline, thus providing an improvement formerly from the first iterations. The objective function found with the MFEI reaches the lowest value of

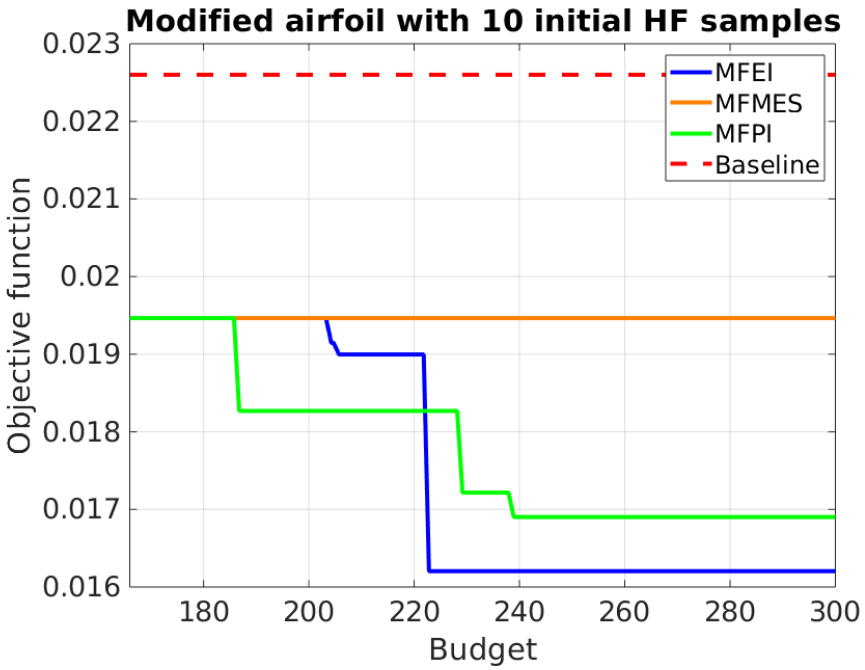


Figure 6.11: Comparison of MFEI,MFMES,MFPI on the modified airfoil with 10 HF and 250 LF initial samples

C_d, 0.0162, by using 129 HF samples and 372 LF (including the initial sampling), whereas that one found with the MFPI achieves a minimum value of 0.0169 by spreading 170 and 283 high- and low-fidelity points respectively. MFPI uses about 50 total samples less than the MFEI, while the last employs the greatest amount of total points with respect to all the other methods. The greater performance of the MFEI are also due to the major exploration made through the employment of the low-fidelity model: in high dimensional problems, it allows to improve more the surrogate, before starting the exploitation. MFMES airfoil, whereas, does not show any improvement from the values observed after the initial sampling, that is 0.0195, and employs around 153 and 320 LF patterns.
About the airfoil shape, MFEI obtains a profile with a slender aft with respect to that depicted by the MFPI. A greater trailing edge radius and a lower τ_{85}/c are evaluated with respect to the MFPI airfoil and, so, to the baseline. Due to a weaker shock, the loss of pressure is reduced downstream of the shock wave and a major lift is developed, thus requiring a minor incidence angle to satisfy the constraint on the C_l. For this reason, the efficiency increases. By comparing the magnitude of the real drag coefficient evaluated with SU2 (Table 6.4) and the objective function of Equa-

tion 6.2, is possible to notice that no penalties are applied to the C_d computed over the MFEI and MFPI airfoils, because all the parameters respect the constraints. Whereas, the leading edge radius of the MFMES profile is slightly below the corresponding constraint. However, this geometry has been selected by the optimizer because, also thanks to the improved τ and τ_{85}/c with respect to the baseline, it allows to reduce the C_d. MFMES airfoil respects the constraint on the lift coefficient with a superior angle of attack than the other methods, thus providing more drag, as it is possible to see from the eddy viscosity representation (Fig.6.13f), therefore less efficiency is reached. The geometry identified by the MFMES depicts a slight shape reduction on the upper surface, with respect to the baseline, indeed it shows the smaller leading edge radius; whereas, the lower surface is increased with respect to the baseline and the aft provides a greater lift, thus increasing a diving pitching moment.

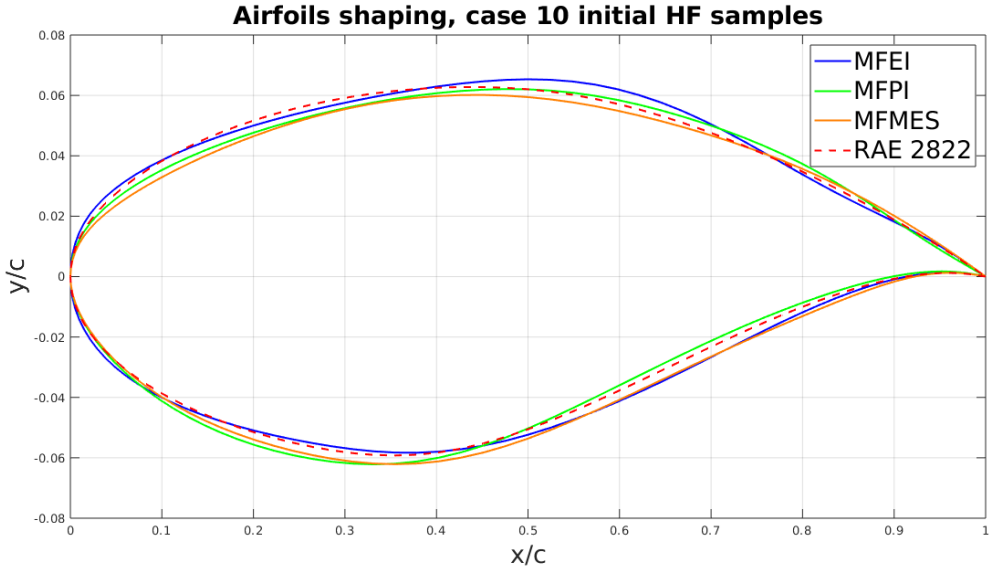
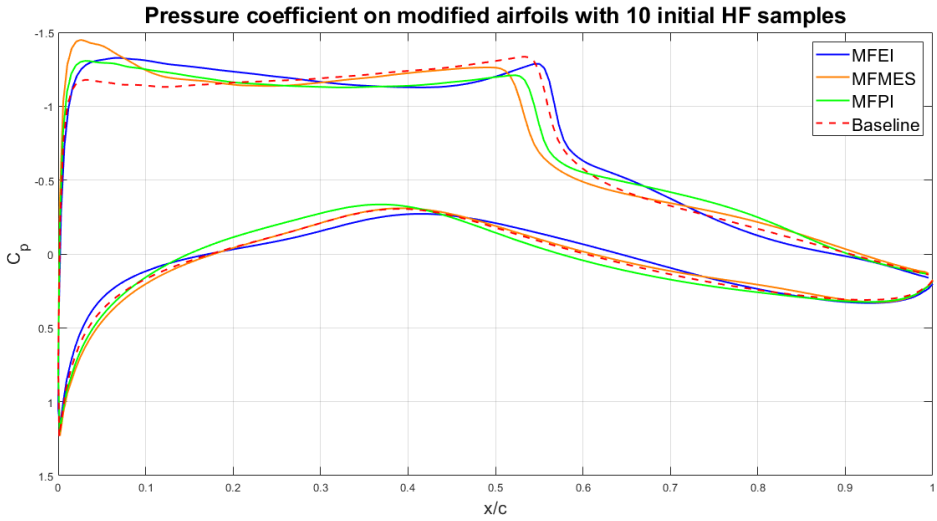


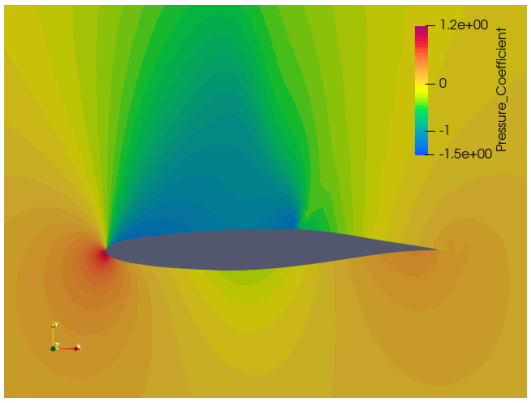
Figure 6.12: Airfoils shaping with MFEI, MFMES, MFPI and 10 initial HF samples

eval_obj.out	**MFEI**	**MFPI**	**MFMES**	**Baseline**
Mach	0.734	0.734	0.734	0.734
α	2.7415°	2.7900°	3.2247°	2.9121°
C_l	0.8239	0.8243	0.8243	0.8239
C_d	0.0162	0.0169	0.0209	0.0226
C_m	-0.0880	-0.0989	-0.0821	-0.0955
Efficiency, $E = C_l/C_d$	50.8096	48.7175	39.2823	36.4742
Max thickness, t/c	0.1211	0.1211	0.1211	0.1211
Leading edge radius, r	0.0112c	0.0074c	0.0063c	0.0079c
Trailing edge angle, τ	9.1072°	6.2157°	9.2562°	8.6808°
Thickness at 0.85c, τ_{85}/c	0.0313	0.0323	0.0354	0.0318

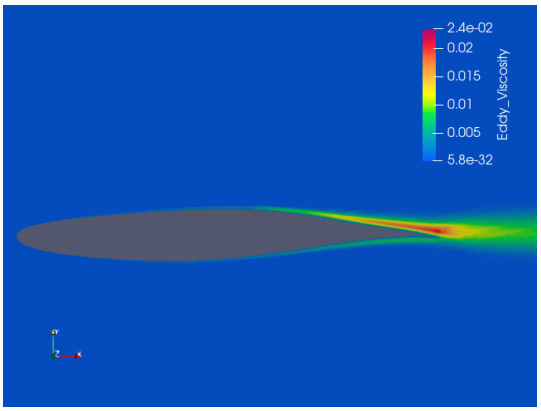
Table 6.4: Output file of the analysis on modified airfoil obtained with MFEI, MFPI, MFMES and 10 initial HF samples.

The pressure coefficient distribution shows the greatest peak at the leading edge, that means a strong acceleration, inducing an anticipation and a more intense shock wave at around the 55% of the chord. A greater pressure load is applied to the aft, with respect to the baseline. For what concerns the bottom side, the profile depicts a pressure that substantially retraces the baseline.
Also the Probability of Improvement finds an airfoil that moves slightly backward the peak of pressure, towards the leading edge, but with a smaller magnitude. It anticipates less the location of the shock wave, with respect to the MES profile, because of a smoother pressure recovery on the upper surface. It provides a greater pressure load on the aft and a major expansion on the bottom side, than the other geometries, due to its greater curvature downward at about the 35% of the chord.
The Expected Improvement delays, but augments, the peak of pressure at the leading edge with respect to the baseline. Then provides a greater compression on the upper surface, and extends the shock wave towards the 60% of the chords, slightly more aft than the original RAE 2822. This is the location of the bump over the foil, that allows to reduce the wave drag: the nearly normal shock wave of the baseline may be replaced by a near lambda wave. It is possible to notice how the pressure coefficient drops less rapidly downstream, because the bump can smear the strong initial shock wave. Hence, a considerable eddy viscosity reduction can be seen with respect to the MES profile, even if both the profiles show a slight anticipation of the region in which the boundary layer starts to thicken, with respect to what can be seen in the MFPI geometry. The multifidelity Expected Improvement airfoil reduces the load on the aft. On the bottom side, there is a smaller rise of pressure at the leading edge. It then increases near the 30% of the chord.
Figure 6.14 summarises what just said: MFEI provides a profile that has a wider region in which the Mach increases, on the top side of the airfoil, and an acceleration zone on the bottom side, which is moved more afterward; MFPI depicts a foil which anticipates the shock wave on the upper surface, and the expansion area over the lower surface; MFMES narrows towards the leading edge the surface over which the Mach increases.

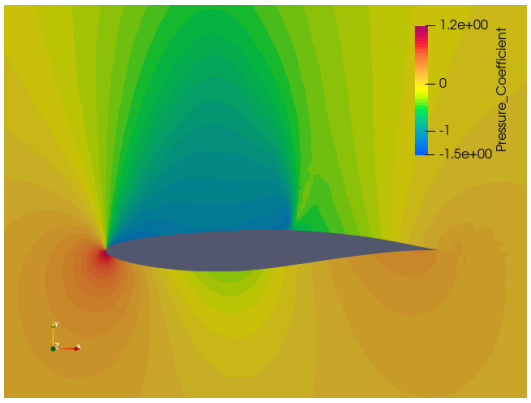




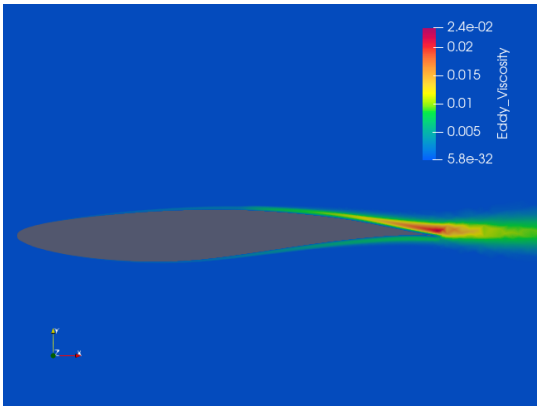
(a) C_p with MFEI



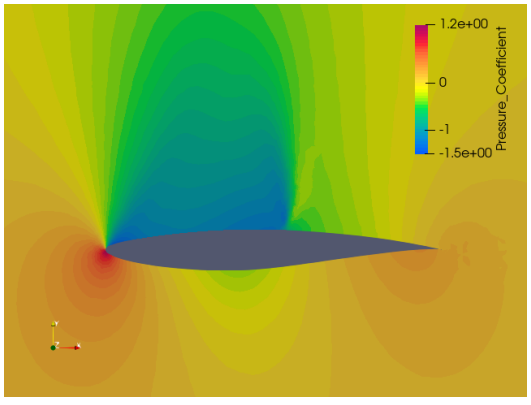
(b) Eddy viscosity μ_t with MFEI



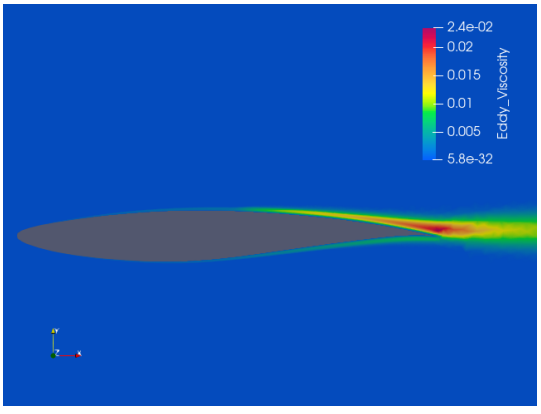
(c) C_p with MFPI



(d) Eddy viscosity μ_t with MFPI

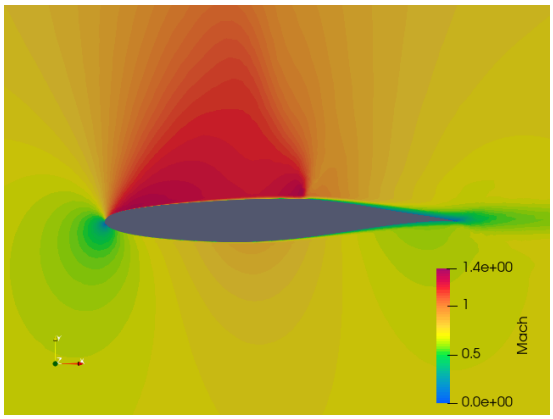


(e) C_p with MFMES

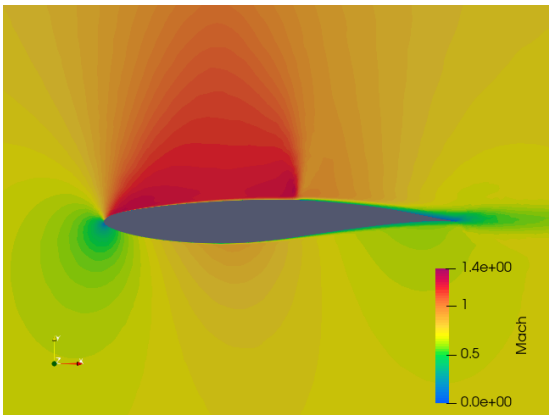


(f) Eddy viscosity μ_t with MFMES

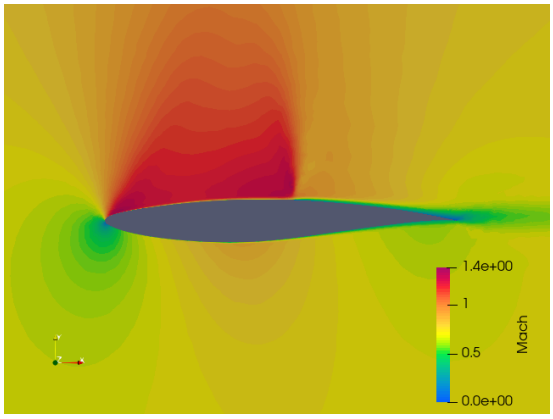
Figure 6.13: C_p and μ_t comparison, over the modified airfoil with 10 HF and 250 LF initial samples.



(a) Mach number with MFEI



(b) Mach number with MFPI



(c) Mach number with MFMES

Figure 6.14: Mach number comparison, over the modified airfoil with 10 HF and 250 LF initial samples.

In the test case which employs 50 HF and 250 LF initial samples, the method that provides the best result is the Multifidelity Probability of Improvement, which can reduce the C_d till a value of 0.0151 by using 179 HF patterns and 262 LF ones. The Expected Improvement, whereas, employs 166 high- and 292 low- fidelity samples to reach a minimum value of 0.0178. The Multifidelity Max-value Entropy Search, provides a slight, but not satisfying, improvement achieving a value of 0.0182 with respect to the initial observation value of 0.0183 and employs 177 and 269 high- and low fidelity function calls. It leads to conclude that the MFMES method requires a greater amount of initial high-fidelity samples to be efficient, but the magnitude of this set of samples needs to be evaluated accurately, because extending excessively the initial sampling contrasts the aim of reducing the expensive HF observations.

For what concerns the shape modeling, PI provides the largest leading edge, while

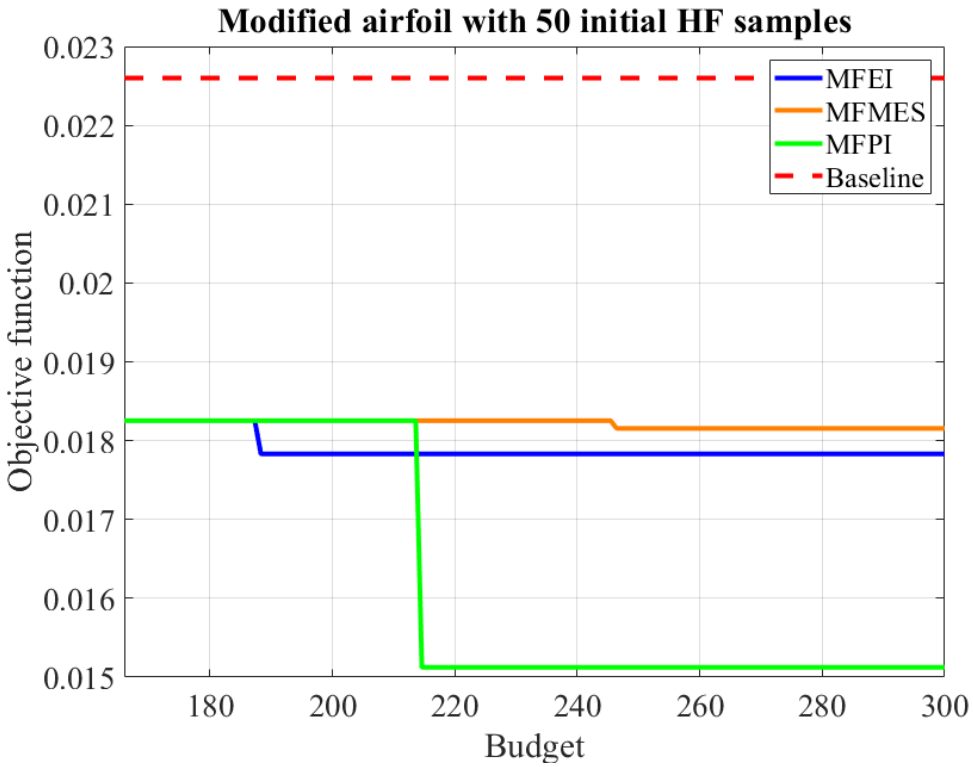


Figure 6.15: Comparison of MFEI, MFMES, MFPI on the modified airfoil with 50 HF and 250 LF initial samples.

the MFEI the narrowest one. MFPI depicts a bump on the upper surface and a thickening on the lower surface of the profile, approximately near the 60% of the chord: it reaches a $\tau_{85}/c = 0.0335$. While the MES tends broadly to follow the baseline, the Expected Improvement computes an airfoil with a larger extension downward of the bottom side, near the 30% of the chord. The geometries selected by the optimizer respect the constraints, so the real C_d and the objective function, computed following Equation 6.2, coincide.

Analysing the pressure distribution over the airfoils, it is possible to notice that all the profiles present a higher acceleration at the leading edge than the baseline, especially the MFEI profile, which also anticipates it. Then, a pressure increase is showed, due to adverse pressure gradients, thus resulting in a shock wave anticipation for all the three geometries. All the airfoils demonstrate a reduction of the

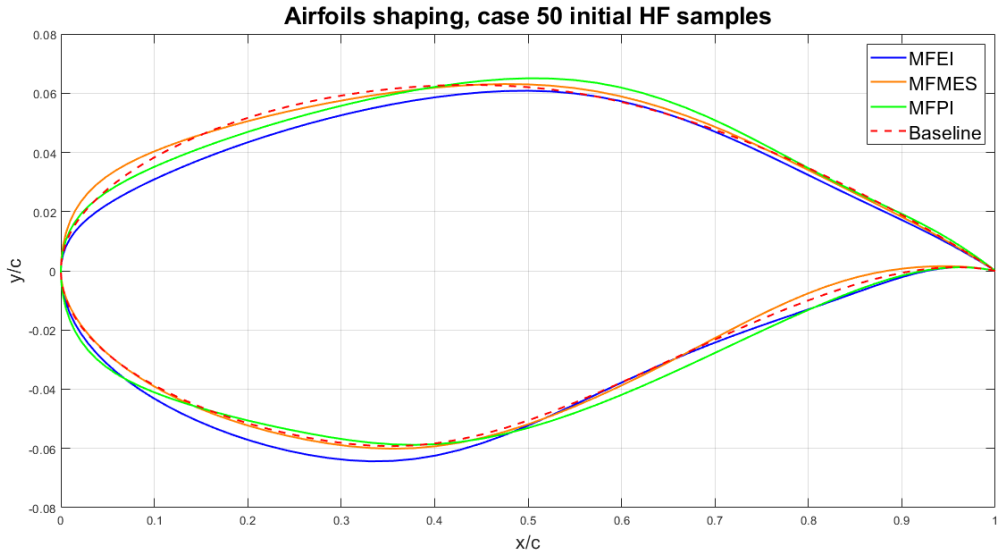
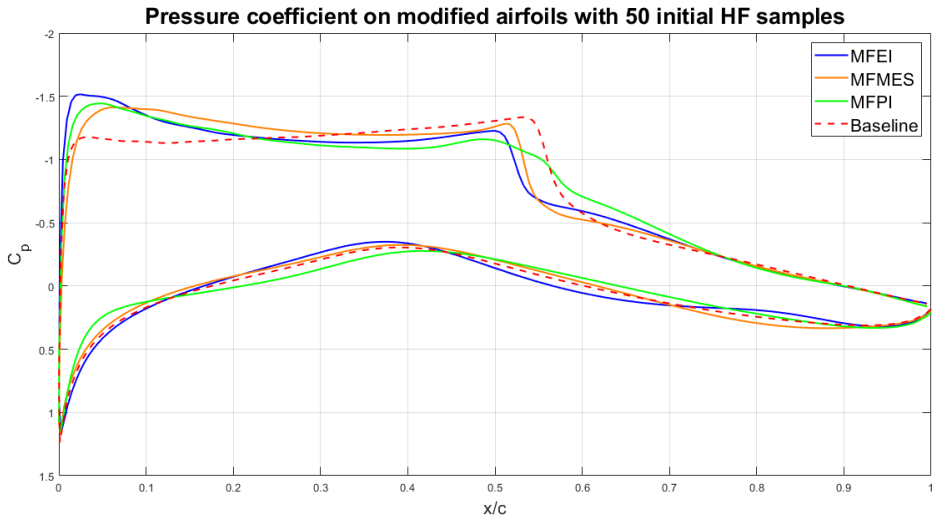


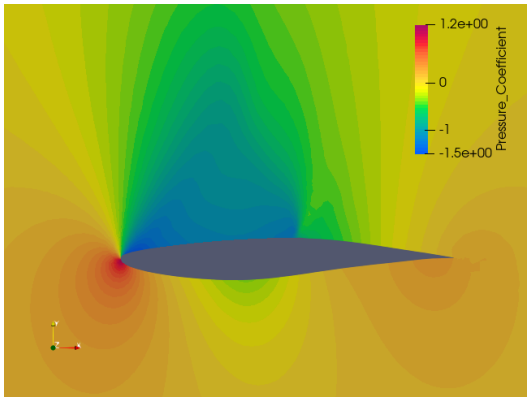
Figure 6.16: Airfoils shaping with MFEI, MFMES, MFPI and 50 initial HF samples

eval_obj.out	**MFEI**	**MFPI**	**MFMES**	**Baseline**
Mach	0.734	0.734	0.734	0.734
α	3.2274°	2.8452°	3.2247°	2.5895°
C_l	0.8241	0.8239	0.8243	0.8241
C_d	0.0177	0.0153	0.0181	0.0226
C_m	-0.0784	-0.0856	-0.0821	-0.0869
Efficiency, $E = C_l/C_d$	46.4817	53.8781	39.2823	45.454
Max thickness, t/c	0.1211	0.1211	0.1211	0.1211
Leading edge radius, r	0.0073c	0.0110c	0.0084c	0.0079c
Trailing edge angle, τ	7.7695°	10.0508°	8.3629°	8.6808°
Thickness at 0.85c, τ_{85}/c	0.0321	0.0335	0.0285	0.0318

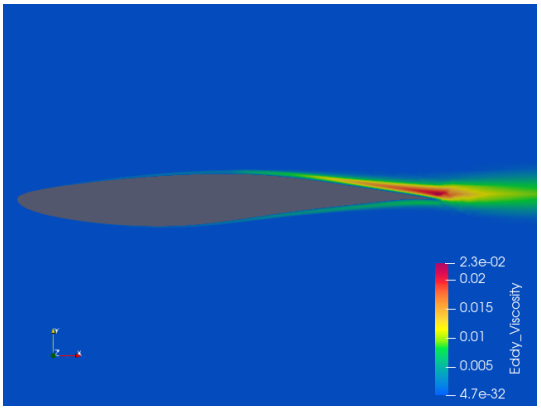
Table 6.5: Output file of the analysis on modified airfoil obtained with MFEI, MFPI, MFMES and 50 initial HF samples.

pressure leap with respect to the baseline, hence the shock produced is weaker. A greater improvement is introduced by the MFPI profile: maybe a lambda shock wave (in-depth analysis could be useful to confirm this hypothesis) is produced, near the bump, that allows to reduce efficiently the drag over the profile, because of its major weakness. The pressure leap after the shock, near the 60% of the chord, results particularly smooth. By observing the Mach distribution in Figure 6.18, the difference among the airfoils result more evident: the profile depicted by the MFMES shows supersonic Mach numbers more forward on the upper surface than the others, thus reflecting the major pressure gradient of a stronger shock wave than the other profiles.

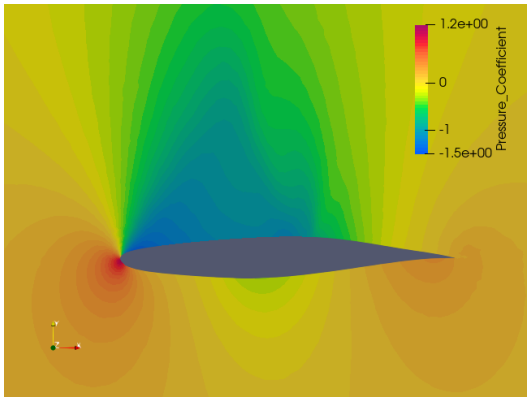




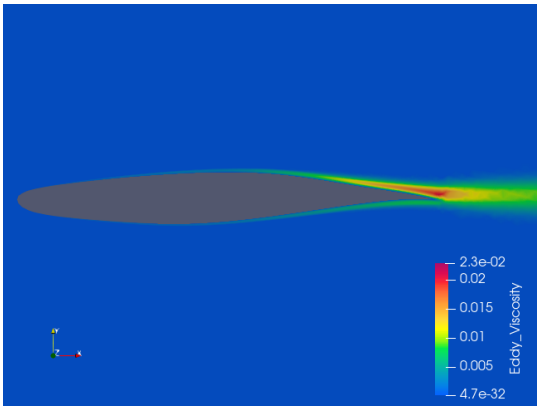
(a) C_p with MFEI



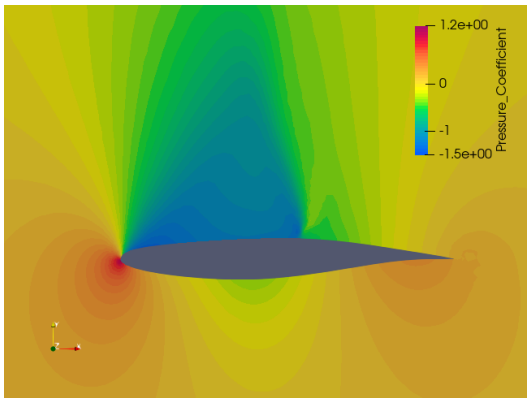
(b) Eddy viscosity μ_t with MFEI



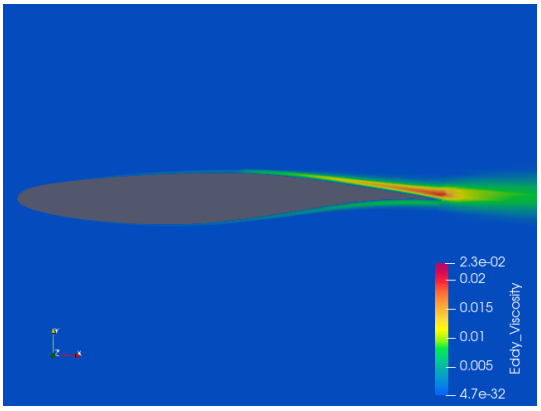
(c) C_p with MFPI



(d) Eddy viscosity μ_t with MFPI

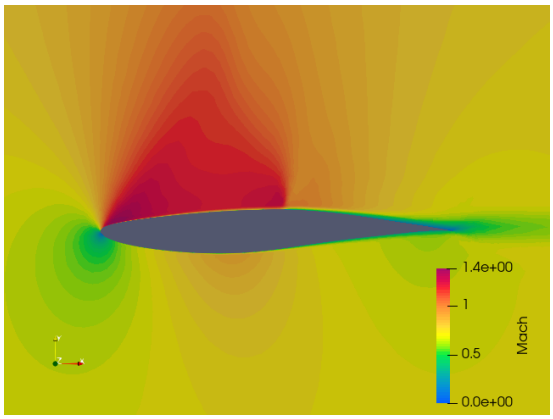


(e) C_p with MFMES

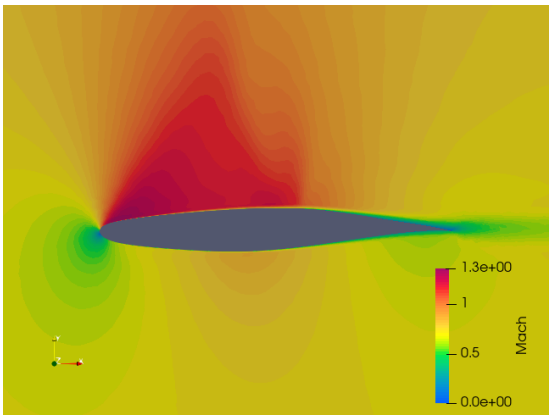


(f) Eddy viscosity μ_t with MFMES

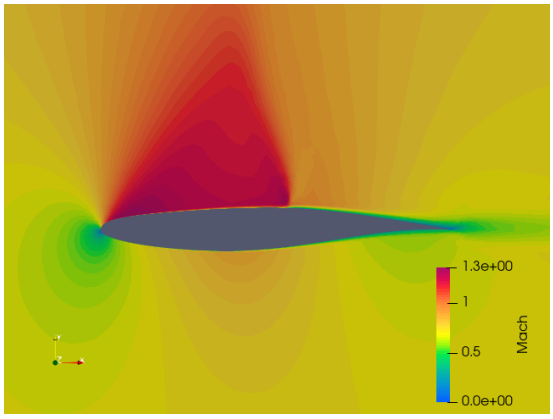
Figure 6.17: C_p and μ_t comparison, over the modified airfoil with 50 HF and 250 LF initial samples.



(a) Mach number with MFEI



(b) Mach number with MFPI



(c) Mach number with MFMES

Figure 6.18: Mach number comparison, over the modified airfoil with 50 HF and 250 LF initial samples.

Therefore, by comparing the two test cases it is possible to observe that MFEI provides useful improvements of the objective function: it can reduce the baseline till 28%, with the narrow initial dataset and till 22% for the extended one. MFPI shows the best performance when the initial set of HF points is increased, providing a reduction of around 32%, with respect to 25% obtained with a smaller set. For what concerns MFMES, in the first test case it allows to decrease the objective function of only 7,5%, while in the last case it can reduce the C_d till 20%.

Chapter 7

Conclusions

The aim of this thesis is to implement and assess three multifidelity Bayesian methods. They employ a surrogate model and a particular acquisition function. The first has been previously built in the context of multifidelity surrogate based optimization, and is iteratively trained via active learning techniques; the latter guide the surrogate through the identification of the next points to query. The surrogate we use combines information collected with just two different fidelity models, while three different acquisition functions distinguishing the methods employed: Multi-Fidelity Expected Improvement (MFEI), Multi-Fidelity Probability of Improvement (MFPI) and Multi-fidelity Max-value Entropy Search (MFMES).
For demonstration purposes, the methods have been applied to simplified problems that trace real aerospace engineering design challenges. Those are benchmarks problems of L1 and L2 class, determined by an international research group, to evaluate multifidelity methodologies. The first class represents problems with a well known analytical formulation; the second one includes more complex engineering problems that are closer to real-life applications.
By assessing the methods over the L1 problems, the Multifidelity Expected Improvement results as the method which assures the best performance. Its advantage lies on a first exploration, that allows to improve the accuracy of the surrogate by reducing the initial HF calls, and on the following exploitation. This behaviour allows to reach the optimum also in a higher dimensionality problem like that of the Rosenbrock 5D, where the other methods run out all the budget leaving a greater residual error. Hence, the exploration technique of the MFEI allows to narrow the initial dataset. The Multifidelity Probability of Improvement, generally shows an initial acceleration towards the optimum with respect to the MFEI, especially when a higher set of initial HF samples is used, thus leading to a greater improvement of the surrogate. Therefore, it has a tendency to call the high-fidelity model for exploitation purposes, thus resulting more expensive. Multifidelity Max-value Entropy Search appears as the method with the lowest performance, because it seems to minimize the objective function more slowly than the others. It is due to the Monte Carlo simulations that add stochasticity to the optimization process. The method shows a general tendency to decrease the total amount of samples to query, by using especially the high-fidelity model. After finding the optimum, both the MFEI and the MFMES demonstrate a particular tendency to repeat the same search paths in some functions. In-depth studies can result useful to understand if this behaviour could lead to a better definition of the optimization stopping criteria.

Among the second class of problems, the methods have been tested on an aerodynamic L2 problem, whose aim is to parametrically shape an airfoil that can minimize the drag rise in transonic regime. The high- and the low-fidelity models are discerned for the grid coarseness of the mesh, so that the low-fidelity model provides a reduction of the computational time of around the 46% with respect to the HF one. A first test case employs a narrow initial set of high-fidelity samples. It shows that the amount of samples is excessively small to assure an improvement with the MFMES, while the MFEI confirms to be able to effectively reduce the objective function even with few initial points. Then, it has been selected a HF set five times greater. High performances have been showed by the MFPI that, as seen in the L1 problems, acts to minimize the objective function especially by calling the high-fidelity model; also the MFEI leads to an improvement, but with a greater residual error than the MFPI; the MFMES slightly reduces the objective function, but is not satisfying as the other methods. Increasing the number of initial high-fidelity samples could improve the performance of the methods, but this set needs to be chosen carefully in order to not make the computation excessively expensive.
L1 and L2 problems are preparatory to extend the use of the methods on L3 problems, that are more complex, expensive and near-reality challenging than the previous classes of problems. In particular, the geometry found with the MFPI, that allows to reduce the objective function till the 32% with respect to the original RAE 2822, can be effectively tested in wind tunnel, and then can be inserted in wing and aircraft contexts. Therefore, these methods allow to analyse in wind tunnel only the most promising configurations, avoiding to spend time and resources, during the phase of design, on unfeasible geometries.
Future developments may concern also the study of the surrogate accuracy achieved at the end of the L1 optimization processes, in order to determine thoroughly limits and advantages of the acquisition functions and to define how the quality of the surrogate changes with increasing dimensionality, thus getting slower the optimization algorithm. Furthermore, it may be interesting to evaluate an accurate fitting of the Monte Carlo simulation, required for the MFMES, with respect to the problem dimensionality. This parameter influences the performance of the MFMES in terms of time and computational costs.

Acknowledgement

Ringrazio la mia relatrice, il Dott. Ing. Laura Mainini, per il continuo supporto tecnico e morale e per la provvidenziale abilità di alleggerire anche i momenti più pesanti. Un doveroso ringraziamento va a Francesco di Fiore, per la sua disponibilità, la sua infinita pazienza e il tempo che mi ha dedicato in questi mesi. Vi ringrazio per avermi fatta sentire parte del team, riducendo i chilometri che ci separavano. Inoltre, ringrazio il Dr. Quagliarella per la sua gentilezza e per il suo tempestivo supporto nella fase di installazione e di setup del problema L2.
Infine, desidero ringraziare Alessandro e tutta la mia famiglia, per aver compreso anche i miei silenzi e per l'assiduo sostegno che non mi è mai mancato in questo lungo e, a volte tortuoso, percorso. Vedere con quanto orgoglio mi presentavate come studentessa di ingegneria aerospaziale, mi ha sempre aiutato a spingermi oltre i miei limiti. Siete la mia gioia più grande.

Bibliography

[1] Cuie Yang et al. "Offline data-driven multiobjective optimization: Knowledge transfer between surrogates and generation of final solutions". In: *IEEE Transactions on Evolutionary Computation* 24.3 (2019), pp. 409–423.

[2] Handing Wang, Yaochu Jin, and Jan O Jansen. "Data-driven surrogate-assisted multiobjective evolutionary optimization of a trauma system". In: *IEEE Transactions on Evolutionary Computation* 20.6 (2016), pp. 939–952.

[3] Leifur Leifsson and Slawomir Koziel. "Multi-fidelity design optimization of transonic airfoils using physics-based surrogate modeling and shape-preserving response prediction". In: *Journal of Computational Science* 1.2 (2010), pp. 98–106.

[4] Spyridon G Kontogiannis et al. "A comparison study of two multifidelity methods for aerodynamic optimization". In: *Aerospace Science and Technology* 97 (2020), p. 105592.

[5] Laura Mainini and Paolo Maggiore. "Multidisciplinary integrated framework for the optimal design of a jet aircraft wing". In: *International Journal of Aerospace Engineering* 2012 (2012).

[6] Stephan Lehner et al. "Advanced multidisciplinary optimization techniques for efficient subsonic aircraft design". In: *48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*. 2010, p. 1321.

[7] MG Hutchison et al. "Variable-complexity aerodynamic optimization of a high-speed civil transport wing". In: *Journal of Aircraft* 31.1 (1994), pp. 110–116.

[8] Matthew Kaufman et al. "Variable-complexity response surface approximations for wing structural weight in HSCT design". In: *Computational Mechanics* 18.2 (1996), pp. 112–126.

[9] Anthony Giunta et al. "Variable-complexity response surface aerodynamic design of an HSCT wing". In: *13th Applied Aerodynamics Conference*. 1995, p. 1886.

[10] Vladimir O Balabanov et al. "Reasonable design space approach to response surface approximation". In: *Journal of Aircraft* 36.1 (1999), pp. 308–315.

[11] Seongim Choi, Hyoung Chung, and Juan Alonso. "Design of low-boom supersonic business jet with evolutionary algorithms using adaptive unstructured mesh". In: *45th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics & Materials Conference*. 2004, p. 1758.

[12] Seongim Choi et al. "Multifidelity design optimization of low-boom supersonic jets". In: *Journal of Aircraft* 45.1 (2008), pp. 106–118.

[13] Juan J Alonso and Michael R Colonno. "Multidisciplinary optimization with applications to sonic-boom minimization". In: *Annual review of fluid mechanics* 44 (2012), pp. 505–526.

[14] Parviz MOHAMMAD ZADEH and Mohsen Sayadi. "An efficient aerodynamic shape optimization of blended wing body UAV using multi-fidelity models". In: *Chinese Journal of Aeronautics* 31.6 (2018), pp. 1165–1180.

[15] Ravindra V Tappeta, Somanath Nagendra, and John E Renaud. "A multidisciplinary design optimization approach for high temperature aircraft engine components". In: *Structural optimization* 18.2 (1999), pp. 134–145.

[16] Ranjan Ganguli. "A survey of recent developments in rotorcraft design optimization". In: *Journal of Aircraft* 41.3 (2004), pp. 493–510.

[17] Bryan Glaz, Peretz P Friedmann, and Li Liu. "Helicopter vibration reduction throughout the entire flight envelope using surrogate-based optimization". In: *Journal of the American Helicopter Society* 54.1 (2009), pp. 12007–12007.

[18] Francesco Di Fiore, Paolo Maggiore, and Laura Mainini. "Multifidelity domain-aware learning for the design of re-entry vehicles". In: *Structural and Multidisciplinary Optimization* 64.5 (2021), pp. 3017–3035.

[19] Edmondo Minisci and Massimiliano Vasile. "Robust design of a reentry unmanned space vehicle by multifidelity evolution control". In: *AIAA journal* 51.6 (2013), pp. 1284–1295.

[20] Marcus A Lobbia. "Multidisciplinary design optimization of waverider-derived crew reentry vehicles". In: *Journal of Spacecraft and Rockets* 54.1 (2017), pp. 233–245.

[21] Amirhossein Adami, Mahdi Mortazavi, and Mehran Nosratollahi. "Multidisciplinary design optimization of a deorbit maneuver considering propulsion, TPS, and trajectory". In: *International Journal of Computer Applications* 116.23 (2015).

[22] Nicolas Duranté et al. "Multi-Disciplinary Analysis and Optimisation Approach for the Design of Expendable Launchers". In: *10th AIAA/ISSMO multidisciplinary analysis and optimization conference*. 2004, p. 4441.

[23] Xiaoyu Zhu, Junli Chen, and Zheng H Zhu. "Adaptive learning observer for spacecraft attitude control with actuator fault". In: *Aerospace Science and Technology* 108 (2021), p. 106389.

[24] Alexander IJ Forrester, András Sóbester, and Andy J Keane. "Multi-fidelity optimization via surrogate modelling". In: *Proceedings of the royal society a: mathematical, physical and engineering sciences* 463.2088 (2007), pp. 3251–3269.

[25] Daniel L Clark Jr et al. "Engineering design exploration using locally optimized covariance Kriging". In: *AIAA Journal* 54.10 (2016), pp. 3160–3175.

[26] Jaroslaw Sobieszczanski-Sobieski and Raphael T Haftka. "Multidisciplinary aerospace design optimization: survey of recent developments". In: *Structural optimization* 14.1 (1997), pp. 1–23.

[27] Jeremy Agte et al. "MDO: assessment and direction for advancement—an opinion of one international group". In: *Structural and Multidisciplinary Optimization* 40.1 (2010), pp. 17–33.

[28] Michael Eldred and Daniel Dunlavy. "Formulations for surrogate-based optimization with data fit, multifidelity, and reduced-order models". In: *11th AIAA/ ISSMO multidisciplinary analysis and optimization conference*. 2006, p. 7117.

[29] Benjamin Peherstorfer, Karen Willcox, and Max Gunzburger. "Survey of multifidelity methods in uncertainty propagation, inference, and optimization". In: *Siam Review* 60.3 (2018), pp. 550–591.

[30] M Giselle Fernández-Godino et al. "Review of multi-fidelity models". In: *arXiv preprint arXiv:1609.07196* (2016).

[31] Chad Lieberman, Karen Willcox, and Omar Ghattas. "Parameter and state model reduction for large-scale statistical inverse problems". In: *SIAM Journal on Scientific Computing* 32.5 (2010), pp. 2523–2542.

[32] Alexander IJ Forrester and Andy J Keane. "Recent advances in surrogate-based optimization". In: *Progress in aerospace sciences* 45.1-3 (2009), pp. 50–79.

[33] Max D Morris and Toby J Mitchell. "Exploratory designs for computational experiments". In: *Journal of statistical planning and inference* 43.3 (1995), pp. 381–402.

[34] Slawomir Koziel, David Echeverría Ciaurri, and Leifur Leifsson. "Surrogate-based methods". In: *Computational optimization, methods and algorithms*. Springer, 2011, pp. 33–59.

[35] Jiju Antony. *Design of experiments for engineers and scientists*. Elsevier, 2014.

[36] Douglas C Montgomery. *Design and analysis of experiments*. John wiley & sons, 2017.

[37] Manny Uy and Jacqueline K Telford. "Optimization by design of experiment techniques". In: *2009 IEEE Aerospace conference*. IEEE. 2009, pp. 1–10.

[38] Ronald Aylmer Fisher et al. "The design of experiments." In: *The design of experiments.* 2nd Ed (1937).

[39] George EP Box and Kenneth B Wilson. "On the experimental attainment of optimum conditions". In: *Journal of the Royal Statistical Society*. 1951, pp. 1–45.

[40] Michael D McKay, Richard J Beckman, and William J Conover. "A comparison of three methods for selecting values of input variables in the analysis of output from a computer code". In: *Technometrics* 42.1 (2000), pp. 55–61.

[41] V Roshan Joseph. "Space-filling designs for computer experiments: A review". In: *Quality Engineering* 28.1 (2016), pp. 28–35.

[42] Gregory D Wyss and Kelly H Jorgensen. *A users guide to LHS: Sandias Latin hypercube sampling software*. Tech. rep. Sandia National Lab.(SNL-NM), Albuquerque, NM (United States), 1998.

[43] Anthony Giunta, Steven Wojtkiewicz, and Michael Eldred. "Overview of modern design of experiments methods for computational simulations". In: *41st Aerospace Sciences Meeting and Exhibit*. 2003, p. 649.

[44] Tushar Goel et al. "Ensemble of surrogates". In: *Structural and Multidisciplinary Optimization* 33.3 (2007), pp. 199–216.

[45] Nestor V Queipo et al. “Surrogate-based analysis and optimization”. In: *Progress in aerospace sciences* 41.1 (2005), pp. 1–28.

[46] Vladimir Cherkassky and Filip M Mulier. *Learning from data: concepts, theory, and methods*. John Wiley & Sons, 2007.

[47] Donald R Jones, Matthias Schonlau, and William J Welch. “Efficient global optimization of expensive black-box functions”. In: *Journal of Global optimization* 13.4 (1998), pp. 455–492.

[48] Simon Haykin and N Network. “A comprehensive foundation”. In: *Neural networks* 2.2004 (2004), p. 41.

[49] Eric Brochu, Vlad M Cora, and Nando De Freitas. “A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning”. In: *arXiv preprint arXiv:1012.2599* (2010).

[50] Steven J Fletcher. *Data assimilation for the geosciences: From theory to application*. Elsevier, 2017.

[51] Stefan M Wild, Rommel G Regis, and Christine A Shoemaker. “ORBIT: Optimization by radial basis function interpolation in trust-regions”. In: *SIAM Journal on Scientific Computing* 30.6 (2008), pp. 3197–3219.

[52] Marc C Kennedy and Anthony O’Hagan. “Predicting the output from a complex computer code when fast approximations are available”. In: *Biometrika* 87.1 (2000), pp. 1–13.

[53] Natalia M Alexandrov et al. “Approximation and model management in aerodynamic optimization with variable-fidelity models”. In: *Journal of Aircraft* 38.6 (2001), pp. 1093–1101.

[54] Natalia M Alexandrov et al. “A trust-region framework for managing the use of approximation models in optimization”. In: *Structural optimization* 15.1 (1998), pp. 16–23.

[55] Inseok Park, Hemanth K Amarchinta, and Ramana V Grandhi. “A Bayesian approach for quantification of model uncertainty”. In: *Reliability Engineering & System Safety* 95.7 (2010), pp. 777–785.

[56] Matthias Poloczek, Jialei Wang, and Peter Frazier. “Multi-information source optimization”. In: *Advances in neural information processing systems* 30 (2017).

[57] Seyede Fatemeh Ghoreishi and Douglas Allaire. “Multi-information source constrained Bayesian optimization”. In: *Structural and Multidisciplinary Optimization* 59.3 (2019), pp. 977–991.

[58] Stephen J Leary, Atul Bhaskar, and Andrew J Keane. “Global approximation and optimization using adjoint computational fluid dynamics codes”. In: *AIAA journal* 42.3 (2004), pp. 631–641.

[59] Donald R Jones. “A taxonomy of global optimization methods based on response surfaces”. In: *Journal of global optimization* 21.4 (2001), pp. 345–383.

[60] Andrew J Booker et al. “A rigorous framework for optimization of expensive functions by surrogates”. In: *Structural optimization* 17.1 (1999), pp. 1–13.

[61] N Alexandrov et al. “Optimization with variable-fidelity models applied to wing design”. In: *38th aerospace sciences meeting and exhibit*. 2000, p. 841.

[62] Andrew March and Karen Willcox. “Constrained multifidelity optimization using model calibration”. In: *Structural and Multidisciplinary Optimization* 46.1 (2012), pp. 93–109.

[63] Andrew R Conn, Katya Scheinberg, and Luis N Vicente. *Introduction to derivative -free optimization*. SIAM, 2009.

[64] Andrew R Conn, Katya Scheinberg, and Luís N Vicente. “Global convergence of general derivative-free trust-region algorithms to first-and second-order critical points”. In: *SIAM Journal on Optimization* 20.1 (2009), pp. 387–415.

[65] Stefan M Wild and Christine Shoemaker. “Global convergence of radial basis function trust region derivative-free algorithms”. In: *SIAM Journal on Optimization* 21.3 (2011), pp. 761–781.

[66] Wild Stefan. “Derivative-Free Optimization Algorithms For Computationally Expensive Functions”. In: (2008).

[67] Roberto Vitali, Raphael T Haftka, and Bhavani V Sankar. “Multi-fidelity design of stiffened composite panel with a crack”. In: *Structural and Multidisciplinary Optimization* 23.5 (2002), pp. 347–356.

[68] Tushar Goel, Raphael T Hafkta, and Wei Shyy. “Comparing error estimation measures for polynomial and kriging approximation of noise-free functions”. In: *Structural and Multidisciplinary Optimization* 38.5 (2009), pp. 429–442.

[69] Loic Le Gratiet and Claire Cannamela. “Cokriging-based sequential design strategies using fast cross-validation techniques for multi-fidelity computer codes”. In: *Technometrics* 57.3 (2015), pp. 418–427.

[70] Andy J Keane. “Cokriging for robust design optimization”. In: *AIAA journal* 50.11 (2012), pp. 2351–2364.

[71] Chanyoung Park, Raphael T Haftka, and Nam H Kim. “Remarks on multi-fidelity surrogates”. In: *Structural and Multidisciplinary Optimization* 55.3 (2017), pp. 1029–1050.

[72] Bruno Betrò. “Bayesian methods in global optimization”. In: *Operations Research’91*. Springer, 1992, pp. 16–18.

[73] J Mockus. “On the Bayes methods for seeking the extremal point”. In: *IFAC Proceedings Volumes* 8.1 (1975), pp. 428–431.

[74] Christopher K Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*. Vol. 2. 3. MIT press Cambridge, MA, 2006.

[75] B.Matérn. *Spatial Variation*. Vol. 2. 1960.

[76] Michael L Stein. *Interpolation of spatial data: some theory for kriging*. Springer Science & Business Media, 1999.

[77] José Miguel Hernández-Lobato, Matthew W Hoffman, and Zoubin Ghahramani. “Predictive entropy search for efficient global optimization of black-box functions”. In: *arXiv preprint arXiv:1406.2541* (2014).

[78] Philipp Hennig and Christian J Schuler. “Entropy Search for Information-Efficient Global Optimization.” In: *Journal of Machine Learning Research* 13.6 (2012).

[79] José Miguel Hernández-Lobato et al. “A general framework for constrained bayesian optimization using information-based search”. In: (2016).

[80] Harold J Kushner. “A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise”. In: (1964).

[81] Zi Wang and Stefanie Jegelka. “Max-value entropy search for efficient Bayesian optimization”. In: *International Conference on Machine Learning*. PMLR. 2017, pp. 3627–3635.

[82] Ronald Aylmer Fisher. “The genetical theory of natural selection: a complete variorum edition”. In: (1930).

[83] Shion Takeno et al. “Multi-fidelity Bayesian optimization with max-value entropy search and its parallelization”. In: *International Conference on Machine Learning*. PMLR. 2020, pp. 9334–9345.

[84] Deng Huang et al. “Sequential kriging optimization using multiple-fidelity evaluations”. In: *Structural and Multidisciplinary Optimization* 32.5 (2006), pp. 369–382.

[85] Xiongfeng Ruan et al. “Variable-fidelity probability of improvement method for efficient global optimization of expensive black-box problems”. In: *Structural and Multidisciplinary Optimization* 62.6 (2020), pp. 3021–3052.

[86] Yixin Liu et al. “Sequential optimization using multi-level cokriging and extended expected improvement criterion”. In: *Structural and Multidisciplinary Optimization* 58.3 (2018), pp. 1155–1173.

[87] Yehong Zhang et al. “Information-based multi-fidelity Bayesian optimization”. In: *NIPS Workshop on Bayesian Optimization*. 2017.

[88] Michalowicz J. *Handbook of Differential Entropy*. Chapman and Hall/CRC, New York, 2014.

[89] Domenico Quagliarella and Matteo Diez. “An Open-Source Aerodynamic Framework for Benchmarking Multi-Fidelity Methods”. In: *AIAA AVIATION 2020 FORUM*. 2020, p. 3179.

[90] András Sobester, Alexander Forrester, and Andy Keane. *Engineering design via surrogate modelling: a practical guide*. John Wiley & Sons, 2008.

[91] Kurt Cutajar et al. “Deep gaussian processes for multi-fidelity modeling”. In: *arXiv preprint arXiv:1903.07320* (2019).

[92] HoHo Rosenbrock. “An automatic method for finding the greatest or least value of a function”. In: *The Computer Journal* 3.3 (1960), pp. 175–184.

[93] Markus P Rumpfkeil and Philip S Beran. “Multi-Fidelity, Gradient-enhanced, and Locally Optimized Sparse Polynomial Chaos and Kriging Surrogate Models Applied to Benchmark Problems”. In: *AIAA Scitech 2020 Forum*. 2020, p. 0677.

[94] Dean Edward Bryson. “A unified, multifidelity quasi-newton optimization method with application to aero-structural design”. PhD thesis. University of Dayton, 2017.

[95] Yun-Wei Shang and Yu-Huang Qiu. “A note on the extended Rosenbrock function”. In: *Evolutionary Computation* 14.1 (2006), pp. 119–126.

[96] Handing Wang, Yaochu Jin, and John Doherty. "A generic test suite for evolutionary multifidelity optimization". In: *IEEE Transactions on Evolutionary Computation* 22.6 (2017), pp. 836–850.

[97] Thomas D Economon et al. "SU2: An open-source suite for multiphysics simulation and design". In: *Aiaa Journal* 54.3 (2016), pp. 828–846.

[98] Francisco Palacios et al. "Stanford University Unstructured (SU2): An open-source integrated computational environment for multi-physics simulation and design". In: *AIAA paper* 287 (2013), p. 2013.

[99] Mark Drela. "XFOIL: An analysis and design system for low Reynolds number airfoils". In: *Low Reynolds number aerodynamics*. Springer, 1989, pp. 1–12.

[100] JD Anderson. *Computational Fluid Dynamics: The Basics with Applications*. McGraw-Hill Book Company Europe, 1995.

[101] Giancarlo Alfonsi. "Reynolds-averaged Navier–Stokes equations for turbulence modeling". In: *Applied Mechanics Reviews* 62.4 (2009).

[102] Pietro Catalano and Marcello Amato. "An evaluation of RANS turbulence modelling for aerodynamic applications". In: *Aerospace science and Technology* 7.7 (2003), pp. 493–509.

[103] Philippe Spalart and Steven Allmaras. "A one-equation turbulence model for aerodynamic flows". In: *30th aerospace sciences meeting and exhibit*. 1992, p. 439.

[104] Francisco Palacios et al. "Stanford university unstructured (SU2): Analysis and design technology for turbulent flows". In: *52nd Aerospace Sciences Meeting*. 2014, p. 0243.

[105] Christelle Wervaecke, Héloïse Beaugendre, and Boniface Nkonga. "A fully coupled RANS Spalart–Allmaras SUPG formulation for turbulent compressible flows on stretched-unstructured grids". In: *Computer methods in applied mechanics and engineering* 233 (2012), pp. 109–122.

[106] John David Anderson. *Modern compressible flow: with historical perspective*. Vol. 12. McGraw-Hill New York, 1990.

[107] Mrinal Kaushik. *Theoretical and experimental aerodynamics*. Springer, 2019.

[108] Charles D Harris. *NASA supercritical airfoils: A matrix of family-related airfoils*. Tech. rep. 1990.

[109] Sonia Chalia. "Offset of Shock Location in Supercritical Airfoils". In: *International Research Journal of Engineering and Technology (IRJET)* 6 (2016), pp. 24–29.

[110] Richard T Whitcomb. "Review of NASA supercritical airfoils". In: *International Council of the Aeronautical Sciences Congress*. ICAS PAPER 74-10. 1974.

[111] BS Stratford. "The prediction of separation of the turbulent boundary layer". In: *Journal of fluid mechanics* 5.1 (1959), pp. 1–16.

[112] Jichao Li, Mohamed Amine Bouhlel, and Joaquim RRA Martins. "Data-based approach for fast airfoil analysis and optimization". In: *AIAA Journal* 57.2 (2019), pp. 581–596.