Detection of discontinuities in nonlinear responses according to different numerical parameters in the context of blade-casing interactions

Sophie Denis

▶ **To cite this version:**

Sophie Denis. Detection of discontinuities in nonlinear responses according to different numerical parameters in the context of blade-casing interactions. Mechanical engineering [physics.class-ph]. 2021. dumas-03258195

HAL Id: dumas-03258195

https://dumas.ccsd.cnrs.fr/dumas-03258195

Submitted on 11 Jun 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

University of Liège
Faculty of Applied Sciences
Department of Aerospace and Mechanical engineering

Development of an algorithm for the detection of discontinuities in a design of experiments

Application to nonlinear systems with contact interfaces

Master thesis written by Sophie DENIS in partial fulfilment of the requirements for the degree of Master in Aerospace Engineering

Supervised by
Doctor Florence NYSSEN (ULiège & LAVA)
Professor Alain BATAILLY (LAVA)
Professor Jean-Claude GOLINVAL (ULiège)

Academic year 2020–2021

Abstract

The drive towards reducing aircraft engines' aerodynamic losses has led engineers to reduce the clearances between blade tips and their surrounding casing, thereby increasing the risk of blade/casing contact interactions. Tools to accurately predict subsequent nonlinear dynamics are thus essential at design stage. Design of Experiments and the construction of surrogate models are very effective in studying complex and costly experimental blade/casing devices. Nonetheless, the convergence of surrogate models is linked to the smoothness of the response, evidencing the requirement of new approximation approaches for discontinuous responses. In this context, this report proposes a methodology to localize discontinuities in a 2D Design of Experiments inputs space in order to apply approximation methods on the smooth subdomains subsequently. First, a mesh is constructed on the inputs parameter space using a Delaunay triangulation and gradient-based indicators along with polynomial annihilation are then used as refinement indicators. The proposed methodology makes use of the Support Vector Machine to create a smooth approximation of the discontinuity location. Local improvement of the solution is finally completed. The applicability and the robustness of the proposed discontinuity localization tool are then tested on a variety of analytical models. The test cases have demonstrated satisfactory localization of closed discontinuities, multiple discontinuities in the same model, and discontinuities running partially through the domain. The numerically costly engineering model of blade/casing contact interactions is then considered and the proposed methodology is used to localize a jump in the response.

Acknowledgments

Throughout the writing of this Master Thesis, I have received a great deal of support and assistance.

First of all, my special thanks are extended to my supervisors, Doctor Florence Nyssen and Professor Alain Batailly, for their valuable and constructive suggestions during the development of this research work. Their insightful feedback pushed me to sharpen my thinking, and I had very much appreciated their willingness to give their time so generously despite the distance and the jet lag.

I would also like to acknowledge my academic supervisor, Professor Jean-Claude Golinval, for his enthusiasm for supervising me during this project. In addition, I would like to thank the jury members, Professor Jean-Philippe Ponthot and Professor Ludovic Noels, who agreed to evaluate my work.

I am grateful to my colleagues from my internship at LAVA, for their help and advice. In particular, I would like to thank Juliette and Thibaut for their precious help in the writing of the report. I also want to thank Philippe and Dany for reviewing my report and Pierre, who advised me a lot about the use of *Python*.

Finally, I am particularly grateful for the assistance given by my family and friends. I would like to thank them for their wise counsel and sympathetic ear. In particular, I must express my very profound gratitude to my mother for providing me with unfailing support and continuous encouragement throughout my years of study.

Contents

List of Figures

List of Tables

Introduction

Motivations

The drive towards increased performance and decreased fuel consumption during aircraft engine operation has led engineers to reduce operation clearances between rotating and stationary components [23], thereby increasing the risk of blade/casing interactions. As a result, risks of contact between the blades and the casing have become an important aspect to consider at the design stage. Indeed, in practice, physical changes in the environment, such as pressure or temperature variations, mean that aeronautical structures do not always operate at their design point. These small variations can lead the machine to operate at lower performance conditions and it is of utmost interest to quantify the impact of such uncertainties in the engine characteristics to optimize its design.

Complex experimental tests are costly and can be resolved more easily with the Design of Experiments. Indeed, numerical models usually allow studying systems in a faster and cheaper way than to perform lots of experiments to study the response of the system to various input parameters. However, model-based technologies can become computationally expensive when the influence of numerous input parameters is studied. For that reason, a reliable method aims to construct a surrogate model with only a limited set of evaluated points. Nonetheless, blade/casing contact interactions are characterized by nonlinear behaviors, exhibiting characteristic nonlinear phenomena such as jumps, bifurcations, tipping points, etc [52]. Such complex discontinuous responses present limitations when using the usual model approximation methods designed for uncertainty quantification, such as Polynomial Chaos expansion [75]. Indeed, the convergence of these stochastic methods is related to the smoothness of the function, and the convergence of the surrogate model changes from exponential to algebraic when discontinuity appears in the response [44]. In particular, Gibbs oscillations could occur in the vicinity of the jump [30], which is, usually, the operating area of interest for the optimization of the components.

To tackle this limitation of usual approximation methods, new approaches for discontinuous responses have been proposed in the literature in the last years, not in the context of blade/casing interaction, but notably in fluid dynamics applications [36]. Three main approaches have been proposed, namely local expanding [72, 45], global expanding [17] and edge tracking [33, 36]. Nevertheless, this research topic is not yet mature. Therefore, approximation methods of discontinuous functions require further research.

Goals

The main objective of this work is the development of an algorithm that detects discontinuities in Design of Experiments. The discontinuities location in the Design of Experiment inputs domain can then be subsequently used to subdivide the domain into smooth subdomains, where the usual polynomial chaos expansions are then constructed. This is the main point of edge tracking methods [33, 36].

The two main challenges of the discontinuity localization tool to develop are: 1) the generalization of the method to various geometries and multiple discontinuities and 2) the localization of discontinuities with only a limited set of evaluation points. The proposed approach relies on some methods that have already been thought of and suggests improvements and novelties over existing methods found in the literature.

Thesis outline

The first chapter of this work is dedicated to the description of the context of the internship in the LAVA research laboratory and the research topic. The second part summarizes the relevant background material on the approximation methods for a discontinuous response. The third chapter constitutes the core of the work. It is devoted to the description of the proposed jump localization method, based on an illustrative example. The influence of some numerical parameters of the method is then discussed. Finally, in the last chapter, the method is applied to various functions to illustrate the robustness of the localization procedure. In particular, the proposed method is tested on an industrial application related to blade/casing contact interactions.

1 | Context of the internship

The Laboratory for Acoustics and Vibration Analysis (LAVA) [1] is a research laboratory under the direction of Professor Alain Batailly, professor at Ecole Polytechnique de Montréal. The laboratory's mission is to investigate noise and vibration issues in advanced technology sectors, including aviation. LAVA's multidisciplinary research work is carried out in collaboration with numerous industrial partners and laboratories.

My role at the LAVA was to conduct research in the context of blade/casing contact interactions. In particular, I was expected to develop an algorithm for the detection of discontinuities in a Design of Experiments, considering the application of nonlinear blade/casing interactions. Two other interns previously performed research on the same topic in 2018. Peschoux [54] and Barbeau's [10] works have been helpful in guiding the research and exploring interesting tools. My research was conducted in relation to a Ph.D. student of the laboratory who is studying polynomial chaos expansion on irregular domains.

The first section aims to provide a review of the blade/casing contact interactions and associated risks to the engine. Since the response is intrinsically nonlinear, the complexity and the main characteristics of nonlinear responses are highlighted. This way, the main challenges of surrogate models construction in this specific context are stated to define the main objectives of the proposed discontinuity detection tool.

1.1 Research topic

The reduction of parasitic leakage flow is a key challenge for aircraft engine design. Indeed, leakage directly influences the flow compression pressure ratio between two stages of an engine [47]. Researchers have thus taken a key interest in components optimization to design engines having a high compression ratio with a small number of stages for weight reduction. The radial distance between the rotating and the stationary parts of the engine is called the blade/casing clearance, as illustrated in Figure 1.1. The leakage flow is directly affected by this parameter and a tighter clearance plays a significant role at improving the engine performance [49].

In practice, this clearance reduction increases the risk of blade/casing interactions. Friction between moving and stationary parts has been recognized as one of the main causes of machine failure [53]. While most blade/casing interactions have a small impact on engine

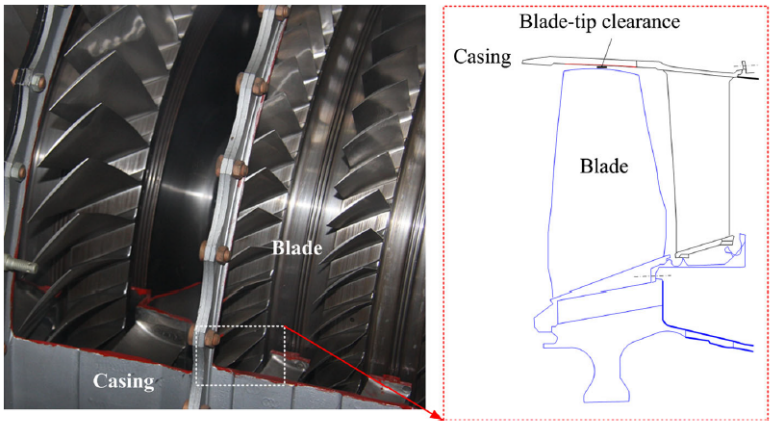


Figure 1.1: Picture and scheme of blade/casing clearance [47].

safety, they are however responsible for some accidents in recent years [47]. In practice, an abradable material is usually added to reduce the vibration amplitude at many angular speeds and reduces the severity of the damages. If contact occurs, the layer peels off, and the surface of the components remains smooth. Nonetheless, contact failure still occurs with this coating [50], which also requires dedicated investigations.

1.2 Characterization of blade/casing interactions

Lots of studies have been carried out to predict contact interactions. These researches have shown that blade/casing contacts are multi-physical, inducing vibrations, rubbing, heating, wear, thermal effects, *etc* [9, 39]. Besides, the studied structure usually has a complex geometry. Nonlinear characteristics of blade/casing interactions have been investigated and demonstrated both experimentally and numerically in recent years [19, 55]. Among others, Cuny et al. [19] present, in their paper, an experimental set-up introduced to study blade/casing interactions of an aircraft engine compressor. Different velocities and separating distances have been experimented, and the results show nonlinear interactions occurring at high speeds. They have shown that contacts may cause cutting, deformation, adhesive transfer, or melting of the components [19].

Besides, Piollet et al. have proposed a test case to serve as a numerical benchmark for blade/casing contact interactions simulations [55]. For that purpose, tests were conducted on the NASA rotor 37, which is a compressor blade widely used in literature of aeronautical studies. The results demonstrate the presence of nonlinear behaviors in blade/casing interactions. Nonlinear features that occur during contact are illustrated in Figure 1.2. The root means square (rms) level of the radial displacements at the Leading Edge of the blade is represented in the case of direct contact (——) and with an abradable coating (——). In direct contact, the rms level increases with the angular speed. Then it decreases

drastically at 1345 rad/s and remains low for higher angular speeds [55]. This jump is characteristic of a nonlinear mechanical system, it characterizes a sudden transition from large to small amplitude of vibrations [52]. Jumps indicate that small variations of the operational parameters drive important change in the response. A second visible nonlinear characteristic is the contribution of odd- and non-integer harmonics. This is shown in Figure 1.3, the spectral map represented shows contribution of odd harmonics at around $1,500$ rpm and $1,900$ rpm [55]. Accounting for the abradable coating wear, nonlinear characteristics such as jump and modal interactions also occur.

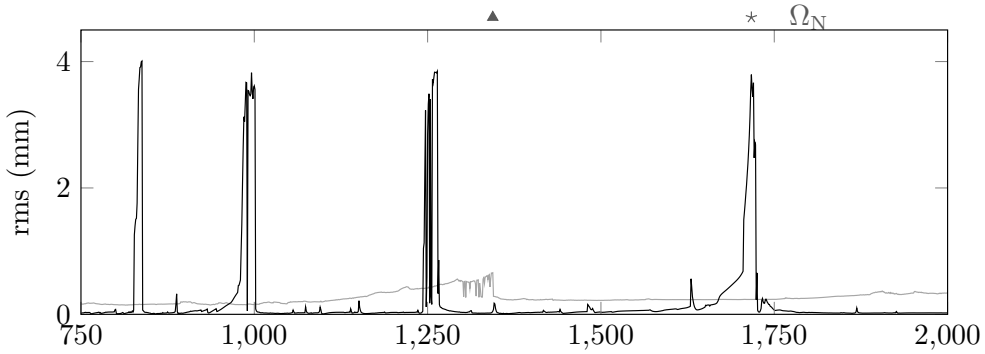


Figure 1.2: Root mean square level of the radial displacements as a function of the rotation speed in the case of direct contact (——) and with an abradable coating (——) [55].

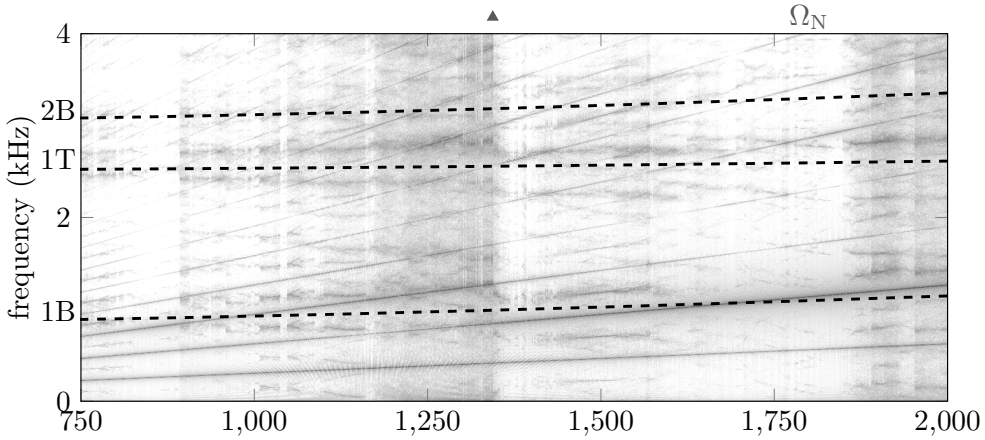


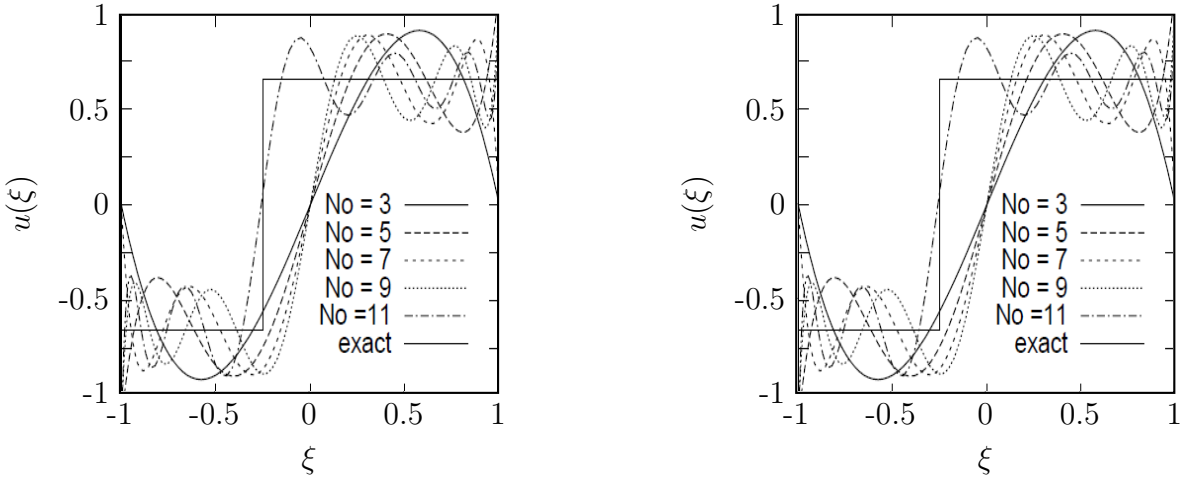
Figure 1.3: Spectral map of blade/casing interactions with direct contact illustrating the evolution of the resonance frequency as a function of the rotation speed [55].

1.3 Uncertainty quantification

Design of Experiments allows studying the response for various inputs parameters. A good characterization of the output requires a very large number of evaluated points, which is a real

problem when dealing with computationally expensive functions. Uncertainty quantification is a useful tool for such complex models. It aims to predict the uncertainties in the response due to the uncertainties in the quantity of interest. Monte Carlo technique is the standard approach, thanks to its relatively easy implementation. Nonetheless, this method is quite inefficient due to its slow-converging property. Spectral expansion provides a better efficiency, reducing the number of evaluation for a good representation of the model [44, 30]. Different methods have been proposed and compared in recent years. These include meta-model or surrogate model [66], reducing significantly computation times [61]. Polynomial chaos has also been used in the context of rotating machines [25, 57]. The goal of this method is to predict the detailed dynamic response in a computationally inexpensive way, but in a sufficient accurate way.

Although the construction of surrogate models have proven to be very effective in many linear model descriptions, the effectiveness of these global response surface approximations drastically decreases when discontinuities are present [40, 60]. The convergence of approximation models is related to the smoothness of the function and it decreases from exponential to algebraic when discontinuity appears in the response [44]. Discontinuities can result in high sensitivity with respect to input uncertainties and oscillatory approximations. For example, the Gibbs phenomenon can occur close to the discontinuities at high order [44, 30], completely destroying the correct approximation of the dynamics. In particular, jump can occur in a zone close to the jump, which is of major interest in the case of components optimization. The bad performances of the pseudo-spectral and the Galerkin approximation methods applied to 1D jump function $u(\boldsymbol{\xi})$ are illustrated in Figure 1.4(a) and (b) respectively, with $\boldsymbol{\xi}$ the input parameter space. As seen, the methods converge



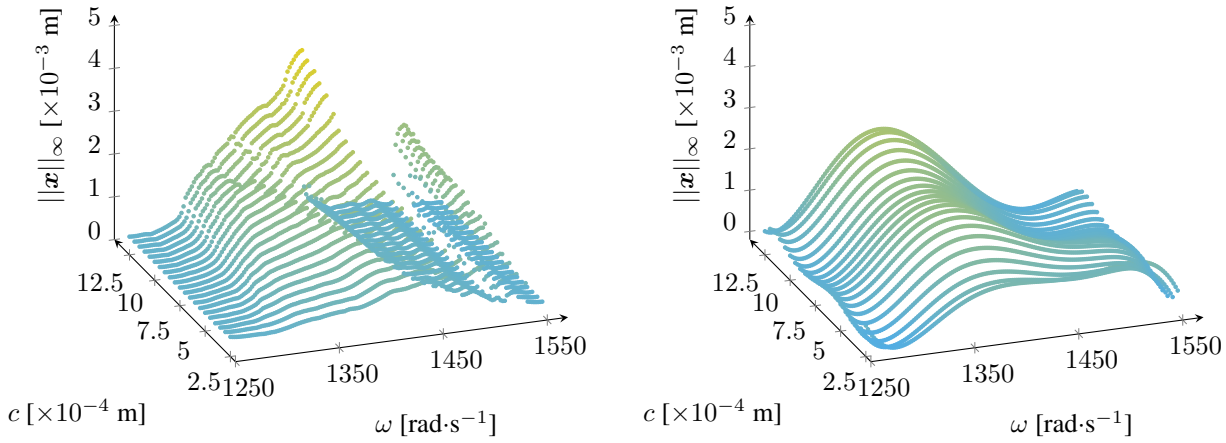
(a) Pseudo-spectral

(b) Galerking

Figure 1.4: Spectral expansion of a 1D non-smooth function $u(\xi)$ showing Gibbs oscillations, with N_o the truncated order [45].

slowly or fail to converge for discontinuous dependence of the solution on the input data since important oscillations appear in smooth parts of the domain on each side of the discontinuity.

Simulations illustrated in Figure 1.5 have been computed by a Ph.D. student in the LAVA, whose work focuses on blade/casing contact interactions. The studied model corresponds to the NASA rotor 37 [58], the industrial application presented in section 4.2. The infinite norm of the blade Leading Edge tip displacement $\|x\|_\infty$ as a function of the engine frequency ω and the clearance c is illustrated in Figure 1.5(a). This response has been obtained evaluating the numerical model on a large number of points. Two jumps are visible in the response introducing discontinuities in the model. On each side of these jumps, the response of the model corresponds to different physical regimes, a small change in the rotation speed can induce a drastic increase in the blade tip displacement and eventually induce contact between the casing and the blade. The application of the polynomial chaos expansion [64] to the model of the blade/casing contact has shown very limited precision on the approximation, as shown in Figure 1.5(b). The largest jump is smoothed, and the second one has completely disappeared.



(a) Deterministic response

(b) Polynomial chaos expansion of degree 5 with 36 deterministic evaluations

Figure 1.5: Blade tip displacement as a function of the rotation speed and the clearance.

The purpose of this work is thus to develop a new methodology, more robust and numerically efficient, to improve the performances of the polynomial chaos method for models presenting jumps and area of steep slope. Different methods have already been developed to improve uncertainty quantification in non-smooth functions. They are described in the following chapter of this work.

2 | Literature review

Different approaches have been developed to make the construction of approximation models applicable when dealing with discontinuous functions and nonlinear behaviors. The three main options developed, namely local approximations, global approximations, and edge tracking are described in this chapter. Some methods have been developed to work only in two dimensions, while more general algorithms may be expanded to higher dimensional spaces.

2.1 Local approximations

A first approach to tackle the problem of the lousy convergence of spectral approximation of non-smooth functions is the local approximation. It consists in dividing the domain into smooth subdomains, where low degree polynomial expansions are applied. This way, local approximations do not require a prior detection of the discontinuities present in the response. Some authors use *general polynomial chaos* (gPC) expansions [72] to discretize the random input space, while others authors use different basis such as *multi-wavelet* functions [45, 62] or *piecewise B-spline* [4].

2.1.1 General Polynomial Chaos expansions

Wan and Karniadakis have developed a method based on the intrusive general Polynomial Chaos (gPC) expansions [72], named the *Multi-Element general polynomial chaos*. The parameter space is divided into smaller elements where gPC is applied again. The decomposition relies on an adaptive process based on the error in variance. This way, the polynomial order degree of the gPC remains small in each subelement, enabling a good convergence of the polynomial expansion.

Initially, the initial set of variables $\boldsymbol{\xi}$ is randomly decomposed in K subsets $\boldsymbol{\xi}^k$, $k = 1, \ldots, K$ and the gPC expansion of one subelement k writes

$$\hat{u}_k\left(\boldsymbol{\xi}^k\right) = \sum_{i=0}^{N_p} \hat{u}_{k,i} \Phi_i\left(\boldsymbol{\xi}^k\right), \tag{2.1}$$

where $\hat{u}_k$ is the approximated field, Φ_i is a basis of multi-dimensional orthogonal polynomials of higher order p and N_p is the total number of basis modes [72]. The local variance σ^2 in

the subelement k is then approximated by

$$\sigma_{k,p}^2 = \sum_{i=1}^{N_p} \hat{u}_{k,i}^2 \left\langle \Phi_i^2 \right\rangle, \tag{2.2}$$

where $\langle \cdot \rangle$ denotes the ensemble average. Besides, the global mean $\bar{u}$ and the global variance $\bar{\sigma}^2$ for all subelements are estimated such as

$$\bar{u} = \sum_{k=1}^{N} \hat{u}_{k,0} J_k \qquad \text{and} \qquad \bar{\sigma}^2 = \sum_{k=1}^{N} \left[\sigma_{k,p}^2 + (\hat{u}_{k,0} - \bar{u})^2 \right] J_k, \tag{2.3}$$

where the J_k parameter is related to the size of the subelement. In addition, the local decay rate of the error of the gPC approximation in each element is defined such as

$$\eta_k = \frac{\sum_{i=N_{p-1}+1}^{N_p} \hat{u}_{k,i}^2 \left\langle \Phi_i^2 \right\rangle}{\sigma_{k,p}^2}. \tag{2.4}$$

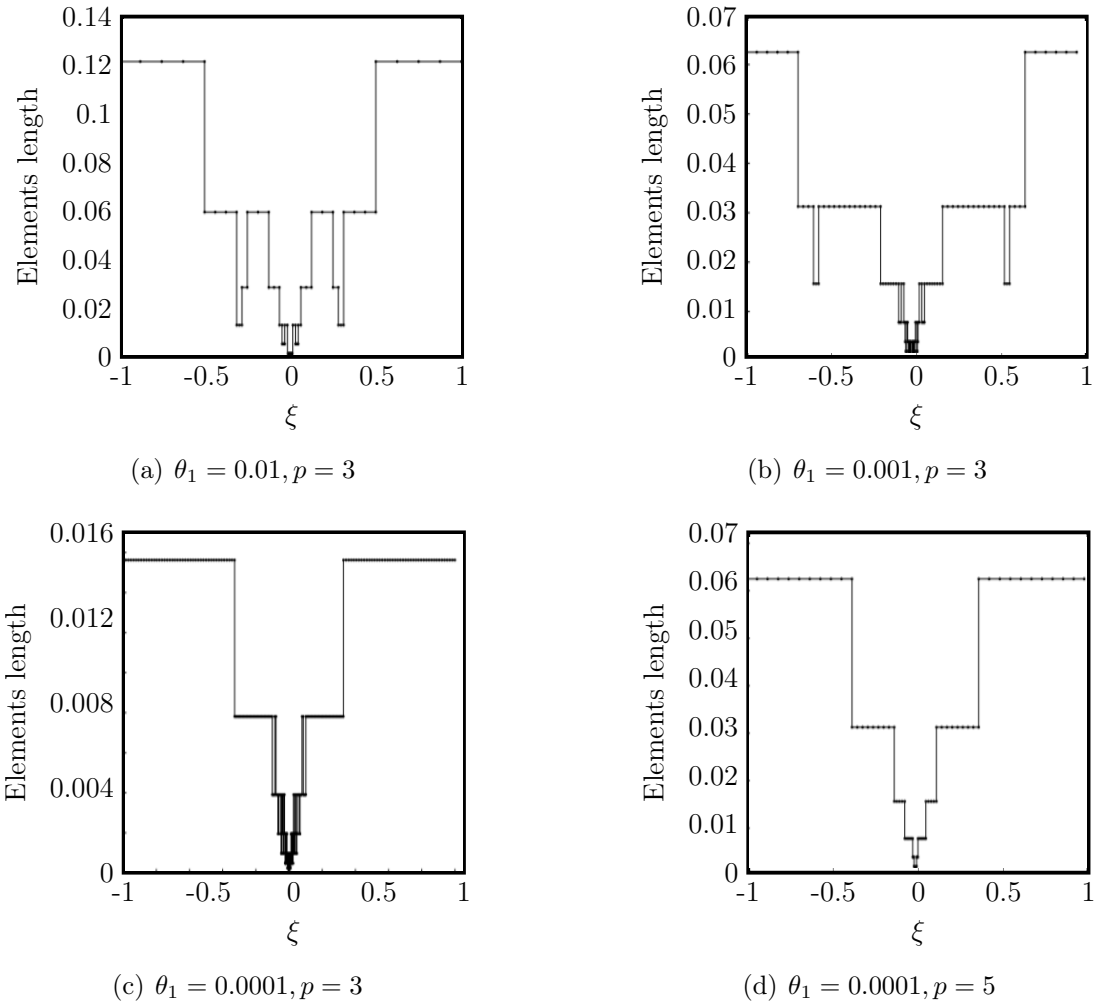
This local decay η_k is used as a benchmark to detect which element must be divided. An element is split into two parts if the conditions defined such as

$$\eta_k^\alpha J_k \geqslant \theta_1, \quad 0 < \alpha < 1 \tag{2.5}$$

is fulfilled. The decay rate of the relative error and factor related to the size of the element must be larger than a prescribed value θ_1, while α is a prescribed constant. For small elements, J_k is small, and the local decay rate of the error must be larger to satisfy the criterion, allowing the division of the domain to stop when the sizes of the subdomain get small.

An example is introduced to illustrate the process in 1D (see Figure 2.1). Nonetheless, the method can be extended in higher dimensions. The step function to be approximated presents a jump at $\xi = 0$. The division of the space is illustrated in Figure 2.1. The elements' length is reduced in the vicinity of the discontinuity. The process has subdivided the initial space into two parts, then into four, where local approximations were constructed. Then the largest variance zones were further refined and so on. As seen, the parameters θ_1 and the order of the polynomial influence the convergence of the method.

The method of Wan and Karniadakis [72] has found applications in many fields. For example, Kewlani *et al.* have used the approach in the context of robotics [43]. It was also used by Le Meitour *et al.* to predict the limit cycle oscillations of airfoils [48]. Multi-Element general polynomial chaos has then been improved by other authors. Chouvion and Sarrouy [18] have modified the error criterion used by Wan and Karniadakis to overcome



(a) $\theta_1 = 0.01, p = 3$

(b) $\theta_1 = 0.001, p = 3$

(c) $\theta_1 = 0.0001, p = 3$

(d) $\theta_1 = 0.0001, p = 5$

Figure 2.1: Size of the mesh in the whole 1D domain [-1;1] for different values of the parameters θ_1 and p [72].

some limitations. Besides, Giovanis and Shields [31] have developed another variance-based approximation method.

To summarize, the local method of Wan and Karniadakis [72] consists of decomposing the domain into smaller non-overlapping subelements. In each subelement, random samples are distributed, and gPC are constructed. The basis used in the local approximation method can vary, but the overall methodology remains the same. Local approximation methods relying on other basis are presented in the following subsection.

2.1.2 Other bases

Le Maître *et al.* have introduced the use of local spectral expansion via the *Wiener-Haar* method [45, 46]. It is a wavelet-based polynomial chaos expansion introduced to limit Gibbs phenomenon and aliasing errors. The approach consists of combining a *multi-wavelet* basis function expansion and a *Galerkin* approach to compute the basis coefficients. A space of

piecewise continuous polynomials is defined, and the function is approximated by using the projection on the space of piecewise polynomials. This way, the expansion based on locally supported bases allows capturing local phenomena and increases the resolution level close to the discontinuity. The method is very robust since bad approximations near the discontinuity do not impact the approximation of the rest of the domain. The Wiener-Haar method was used in chemical structure studies involving strong nonlinear dependence on temperature effect [51]. Nonetheless, it has the drawback of requiring lots of model evaluations.

Schiavazzi *et al.* [62] have also used the multi-wavelet expansion. The main difference with that of Le Maître [45] is that the multi-wavelet coefficients computation relies on a Compressive Sampling strategy rather than Galerkin projections. They applied the approach to cardiovascular modeling [62]. More recently (2020), Abdedou *et al.* [4] have introduced a non intrusive stochastic multi-element approach relying on local approximations of the response using piecewise B-spline basis functions.

2.1.3 Limitations

Although local approximation improves the convergence of approximation of non-smooth functions, the approach presents some limitations. Indeed, even if Gibbs oscillations are centralized in the vicinity of the discontinuity, they still occur, limiting the precision of the approximation. In addition, local approximation requires new model evaluations and quadrature points in each subdomain, which can become computationally costly after numerous domain decompositions. Indeed, since each axis is split and the hypercube is redefined on each subpart of the axis, and since the method requires a tensor structure, the number of model evaluations drastically increase whith the number of grid divisions. Global approximations, discussed just below, have been introduced to reduce computation times compared to local methods.

2.2 Global approximations

The second method to increase the convergence of the approximation model in discontinuous functions are the global approximations. They have the advantage of reducing computation times compared to the local methods. Global methods rely on the observation that discontinuities cause slow decay of the coefficients of spectral expansions.

Chantrasmi *et al.* have developed a global method [17] based on *Padè approximation* of discontinuous functions in the context of fluid dynamics modelization. The domain is evaluated in a deterministic manner on points uniformly distributed on the domain, and a Padè Legendre approximation is constructed as a response surface approximating the exact

model.

By definition, *a Padé approximation is a rational function – a ratio of two polynomials – that can be thought of as a generalization of a Taylor series expansion* [17]. Every function u in the Hilbert space, which is defined on [-1;1] and is square-integrable, can be written with a Legendre polynomial expansion and approximated by the truncated Taylor series expansion, such as

$$u = \sum_{n=0}^{\infty} \hat{u}_n P_n \approx \sum_{n=0}^{N} \hat{u}_n P_n, \tag{2.6}$$

where $\hat{u}_n$ is the nth Legendre coefficient and P_n the Legendre polynomial of degree n, with N the largest degree. The rate of convergence of the approximation is proportional to the smoothness of the model u, such as for discontinuous function, the Padè Legendre expansion converges slowly. This property is the key points of the method, since it allows the localization of the discontinuities. The numerical model u is approximated by a rational function $R(u)$ defined as

$$u \approx R(u) := P/Q, \tag{2.7}$$

where P and Q are linear combination of Legendre polynomial P_j of order M and L respectively

$$P = \sum_{j=0}^{M} \hat{p}_j P_j \qquad \text{and} \qquad Q \ = \sum_{j=0}^{L} \hat{q}_j P_j. \tag{2.8}$$

The oscillating and improper results of the global expansions are used to filter the coefficients and create a smooth approximation of the discontinuity. The coefficients of the polynomial are computed in such a way that $\langle P - Qu, \phi \rangle_N = 0$, where ϕ is chosen as the Legendre polynomials P_n, with $n > M$, and $N = M + L$. The zero of the function Q corresponds to the location of the discontinuity.

The main advantage of this method is the possibility to use the existing global polynomial, which captures the smooth response. Nonetheless, undesirably large degrees of smoothing can be required when dealing with very complex discontinuous engineering models. Due to the limitations of both local and global approximations, an edge tracking method relying on the prior localization of the discontinuity has been introduced. This is the topic of the following section.

2.3 Edge tracking

The third method relies on the decomposition of the parameter space into smaller smooth elements where spectral expansions are constructed individually. The edge detection part is thus carried out upstream of the approximation of the response. An algorithm is thus

required to effectively detect the position of the discontinuities, which are defined as the edge of continuous areas and may be open or closed.

Locating discontinuities is a problem that has been little studied in the aeronautical field. Nonetheless, discontinuity or abrupt changes studies have shown particular interest in many fields, such as image recognition [12], digital cartography [35], environmental sciences [28], population study in biology and ecology [76], disturbance in electrical field [69] *etc.* Most of the algorithms developed in such contexts require the input data to be dense and distributed on a uniform grid. This is not well suited within the frameworks of complex engineering models due to the long computation times.

Nonetheless, edge tracking makes use of some interesting approaches developed to locate discontinuities with a dense distribution of input data. For example, Voronoi tessellations were introduced by Debnath *et al.* [21, 22] to determine the masses of new physics particles, using neighbor characteristics to detect zones of abrupt changes. Besides, Womble [76] has developed the *Wombling* using the gradient between points distributed on the domain and a threshold to detect the zone of discontinuities. This method was adopted by Fortin [26] in the context of detection of ecotones in forests. The latter and Drapeau [27] have introduced the technique of triangulation in the context of jump in numerical data, but their method had the disadvantage of identifying edges in areas of rapid changes. To remedy this restriction, Barbujani *et al.* [11] used statistical significance tests to validate the detection. Gabriel [28] has also developed a method based on local statistic tests of the local gradients to detect discontinuous areas.

Inspired by all these methologies, different methods have been developed to detect discontinuities in a stochastic space. The most interesting ones for the purpose of this work are presented in the next paragraphs, as well as their pros and cons. This part focuses mainly on the discontinuity detection step since the approximation of the response in each smooth subdomain is quite similar for the different authors, and it is not the main purpose of this Master thesis work.

2.3.1 Bayesian inference

Sargsyan *et al.* [61] have developed a discontinuity localization method involving a probabilistic description of the shape of the discontinuity, given a small number of evaluations of the model. Average over all possible discontinuity locations are considered, providing localization of the discontinuity without requiring numerous data nearby the jump. The probabilistic approach relies on *Bayesian inference*. Developments in dimensions larger than 2 are applicable, but the computation times become very large.

In 2D, the method assumes that the model $u(x, y)$ presents a discontinuity along a curve $\mathcal{G}(x, y) = 0$, which is unknown a priori. The model is evaluated at N points (x, y). As a first step, the yet unknown discontinuity curve $\mathcal{G} = 0$ is described as a polynomial function of order d, such as

$$\mathcal{G} := y = p_{\mathbf{c}}(x) = \sum_{k=0}^{d} c_k x^k, \tag{2.9}$$

where c_k are the coefficients of the polynomials.

The goal of the method is to infer the polynomial coefficients c_k. For that purpose, an approximation model $\mathcal{M} = g(x, y)$ is chosen for function u. A hyperbolic tangent function is a good choice that imitates a discontinuity behavior, but other nonlinear functions can be used. In particular, multiple discontinuities can be represented using multiple hyperbolic tangents. Relying on this approximation model $\mathcal{M}$, Bayesian inference is used to infer the polynomial coefficients. The posterior probability of the model $\mathcal{M}$ [61], given N output data $\mathcal{D} = u_1(x_1, y_1), u_2(x_2, y_2), ...u_N(x_N, y_N)$, is written using Bayes's formula such as

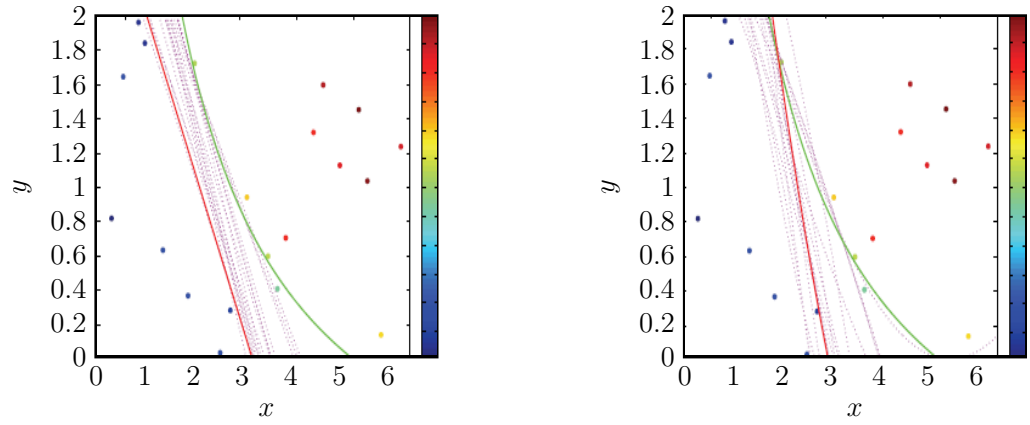
$$P(\mathcal{M} \mid \mathcal{D}) = \frac{P(\mathcal{D} \mid \mathcal{M})P(\mathcal{M})}{P(\mathcal{D})}. \tag{2.10}$$

$P(\mathcal{M})$ represents the degree of belief about the model before having the data (uniform in the absence of no additional information) and $P(\mathcal{M} \mid \mathcal{D})$ after having the data $\mathcal{D}$. The function $P(\mathcal{D} \mid \mathcal{M})$ represents the likelihood to obtain the data set $\mathcal{D}$ if they were computed with the model $\mathcal{M}$. *Markov Chain Monte Carlo sampling* [70] is used to find samples from the posterior distribution $P(\mathcal{M} \mid \mathcal{D})$. The results give a distribution of possible discontinuity curves described by a set of sample parameters vector $\mathbf{c}$. The output is not the exact location of the discontinuity, but possible discontinuity locations used to separate the space into smooth regions.

An example of the probabilistic discontinuity location obtained with the Bayesian inference, considering two different polynomial orders d, is illustrated in Figure 2.2. The left part corresponds to linear inference ($d = 1$), and on the right side, a quadratic order is fixed ($d = 2$). The colored points correspond to the evaluation of the model ($N = 10$ or 100, resp. above and below), as seen they are well distributed in the whole domain and not only in the vicinity of the jump. The true discontinuity curve (——), some samples of the discontinuity (- - - -) obtained from the posterior distribution and the Maximum A Posterior estimation of the position (——) are represented in Figure 2.2. The method gives a quite reliable estimate of the discontinuity location. The results are even better when 100 points are evaluated along with the quadratic polynomial.

The main advantage of this method is the evaluated points distribution that is uniform on

the whole domain. Nonetheless, using Bayesian inference leads to less accurate localization of the discontinuity than the approach presented in the following section. In addition, bad distribution of initial data could miss the detection. The second edge tracking method introduced in the next section presents restrictions on the position of the sampling points but provides more precise localization of the discontinuity.



(a) $d = 1$, $N = 10$ (b) $d = 2$, $N = 10$

(c) $d = 1$, $N = 100$ (d) $d = 2$, $N = 100$

Figure 2.2: 2D discontinuity localization using the Bayesian inference detection tool with polynomials of degree 2 and 3 [61].

2.3.2 Polynomial annihilation

Polynomial annihilation is another edge tracking approach for discontinuity detection that has been introduced by Archibald *et al.* [8]. The method relies on the construction of a jump function based on a set of irregularly distributed points in the domain.

By definition of the discontinuity, a local jump function u is defined, such as

$$[u](x) = u(x^+) - u(x^-), \tag{2.11}$$

where $u(x^+)$ and $u(x^-)$ are the right and left limit of the function at x. For a smooth function, $u(x)$ is equal to zero, while at the discontinuity, the jump function is equal to the value of the jump. The method relies on the construction of an approximation of this jump function, denoted $L_m u$. This approximation function has the property to converge towards zero away from the discontinuity, with a convergence rate proportional to m, a positive integer. The approach can be extended to higher dimensions, but it is presented in a one-dimensional space to simplify the notations.

The illustrative example used is shown in Figure 2.3. For a point x of the domain, a set of points $\mathcal{S}_x$ containing m_d points around x is chosen, with $d = 1$, the dimensions of the space. The approximation of the jump function [8] is constructed such as

$$L_m u(x) = \frac{1}{q_m(x)} \sum_{x_j \in \mathcal{S}_x} c_j(x) u\left(x_j\right). \tag{2.12}$$

A linear system is solved for the coefficients $c_j(x)$, as

$$\sum_{j \in S_x} c_j(x) p_i\left(x_j\right) = \sum_{|\alpha|_1 = m} p_i^{(\alpha)}(x), \quad i = 1, \ldots, m_2, \quad \alpha \in \mathbb{Z}_+^2, \tag{2.13}$$

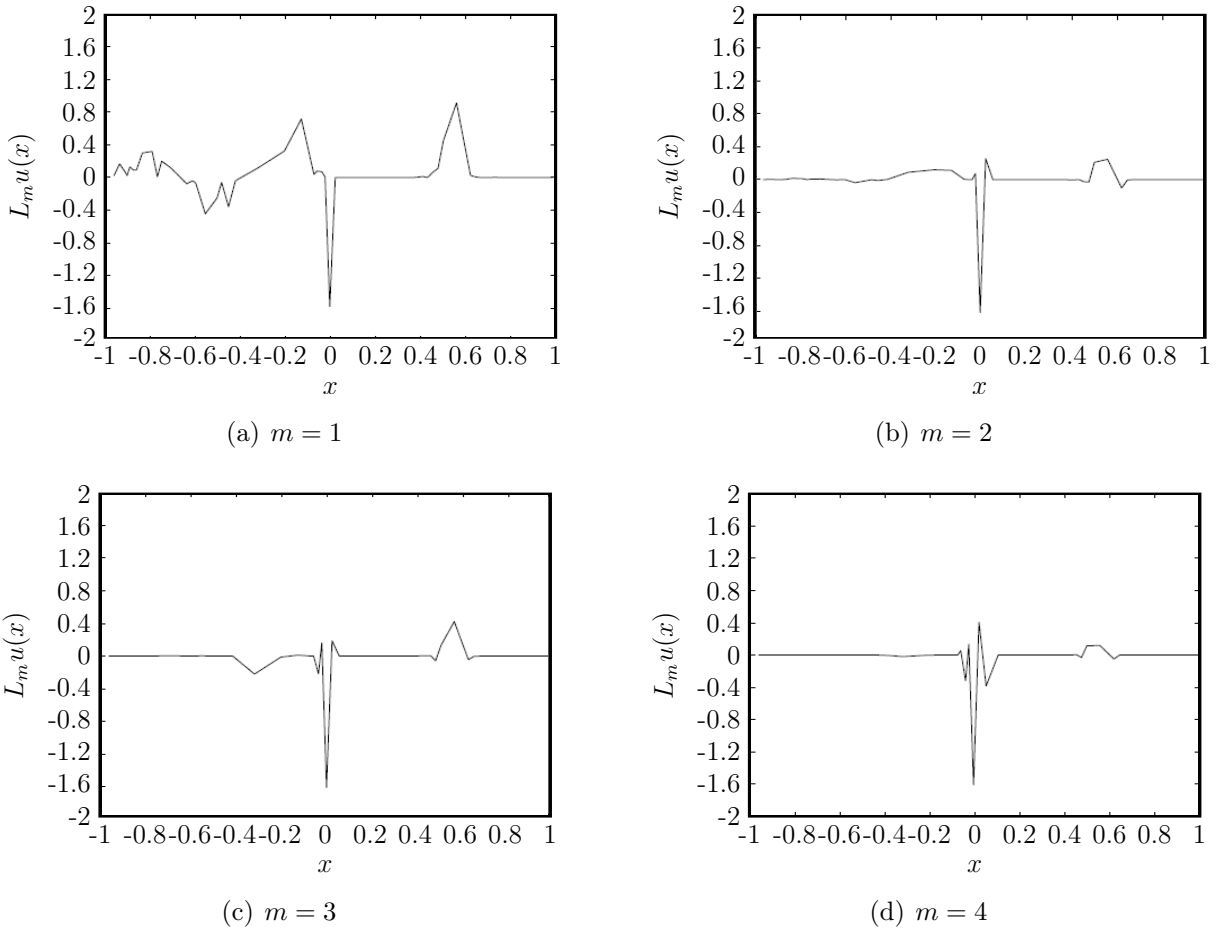
where p_i are polynomial of order i and $\cdot^{(\alpha)}$ denotes the $\alpha-$derivative. The normalization factor q_m is used to estimate the value of the jump in 1D or its orientation in multi-space, it is defined such as

$$q_m(x) := \sum_{x_j \in \mathcal{S}_x^+} c_j(x). \tag{2.14}$$

The convergence rate towards zero away from the discontinuity depends on m and the local smoothness of u. Nonetheless, large values of m will increase oscillation in the vicinity of the jump and reduce the accuracy of the precision. The values of the edge function $L_m u$, for a step function that has a jump of amplitude 2 at $x = 0$, with m ranging from 1 to 4, are illustrated in Figure 2.3. The fourth-order function gives a better accuracy, but oscillations appear close to the jump location.

To remedy this defect, the *min-mod* function is applied. Considering a set of $\mathcal{M}$ positive integers, the highest order of convergence is ensured by using the min-mod function defined as

$$MM\left(L_{\mathcal{M}} u(x)\right) = \begin{cases} \min_{m \in \mathcal{M}} L_m u(x) & \text{if } L_m u(x) > 0 \text{ for all } m \in \mathcal{M}, \\ \max_{m \in \mathcal{M}} L_m u(x) & \text{if } L_m u(x) < 0 \text{ for all } m \in \mathcal{M}, \\ 0 & \text{otherwise .} \end{cases} \tag{2.15}$$



(a) $m = 1$ (b) $m = 2$

(c) $m = 3$ (d) $m = 4$

Figure 2.3: $L_m u(x)$ for different values of m [8].

The improvements in the approximation of the jump function are illustrated in Figure 2.4. The mid-mod limiter combined with the annihilation edge detection shows a reliable detection of the unique jump at $x = 0$.

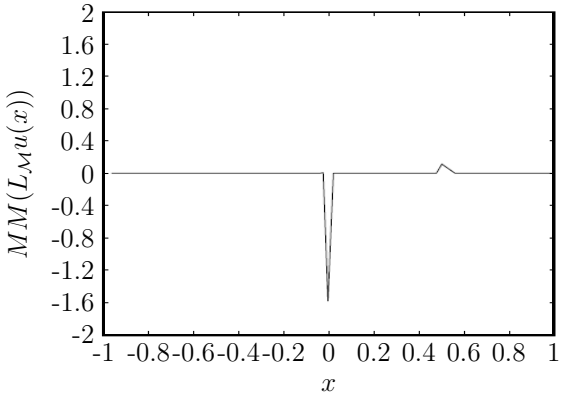
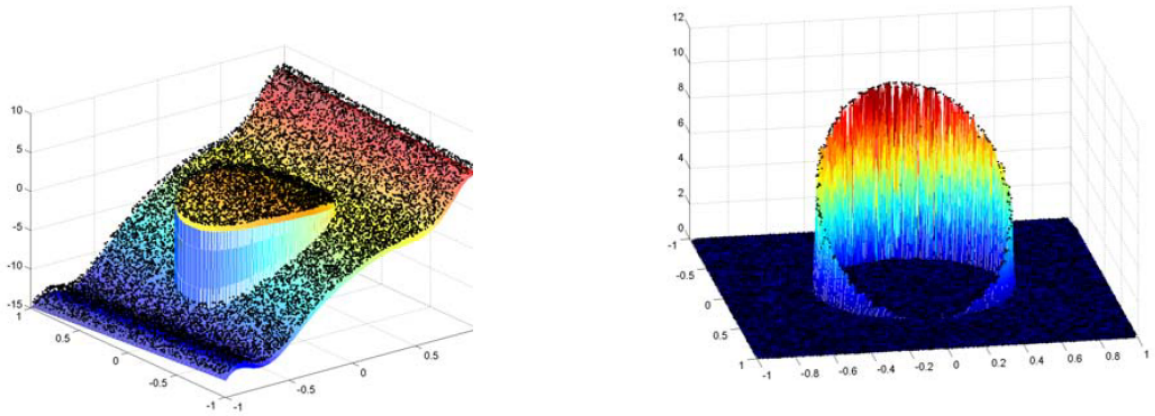


Figure 2.4: The min-mod method with $\mathcal{M} = 6$ [8].

The results of the edge detection method applied to a circular discontinuity in two dimensions are illustrated in Figure 2.5. The evaluated nodes are the vertices of a *Delaunay triangulation* [42], which is a triangulation such that for a given set of points $\mathcal{P}$, no point in $\mathcal{P}$ is inside the circumcircle of any triangle. The main limitation of this method is the requirement, as seen in Figure 2.5, of a drastically high number of grid points. Nonetheless, the method has the advantage to be independent on the geometry of the discontinuity and allows the distinction between steep gradient and discontinuities areas. To reduce the number of grid points, Jakeman *et al.* [41] have introduced the use of an adaptive refinement depending on the value of the jump function.



(a) $u(x, y)$ (b) $L_m(u)$

Figure 2.5: Illustration of the polynomial annihilation edge detection method on a 2D case with a circular discontinuity [8].

2.3.3 Stochastic polynomial annihilation

Archibald *et al.* [7] have introduced an improvement in the polynomial annihilation method in the context of the costly numerical models. A stochastic approximation u_N of the jump function u is used instead of evaluating the true model, which was the most time consuming part in the method previously described [8] and used by Jakeman *et al.* [41].

The jump function is approximated with a Nth order general polynomial chaos such as

$$u_N\left(y^{[\ell]}(b)\right) = \sum_{\mathbf{i}\in\mathcal{J}_N} \hat{u}_{\mathbf{i}}\Phi_{\mathbf{i}}\left(y^{[\ell]}(b)\right), \tag{2.16}$$

where Φ_i is a gPC basis and $y^{[\ell]}(b) = (b^{(1)}, \ldots, b^{(l-1)}, y^{(l)}, b^{(l+1)}, \ldots, b^{(d)}$ with $b = (b^{(1)}, \ldots, b^{(d)})$ a set of fixed coordinates in $[-1; 1]^d$. This way, the approximation of the jump function with

the polynomial annihilation edge detection method writes

$$\begin{aligned} L_m u_N\left(y^{(\ell)}\right) &= \frac{1}{q_m\left(y^{(\ell)}\right)} \sum_{y_j^{[\ell]}(b)\in S^{(\ell)}(b)} c_j\left(y^{(\ell)}\right) u_N\left(y_j^{[\ell]}(b)\right) \\ &= \frac{1}{q_m\left(y^{(\ell)}\right)} \sum_{y_j^{[\ell]}(b)\in S^{(\ell)}(b)} c_j\left(y^{(\ell)}\right) \sum_{\mathbf{i}\in\mathcal{J}_N} \hat{u}_{\mathbf{i}} \Phi_{\mathbf{i}}\left(y_j^{[\ell]}(b)\right), \end{aligned} \tag{2.17}$$

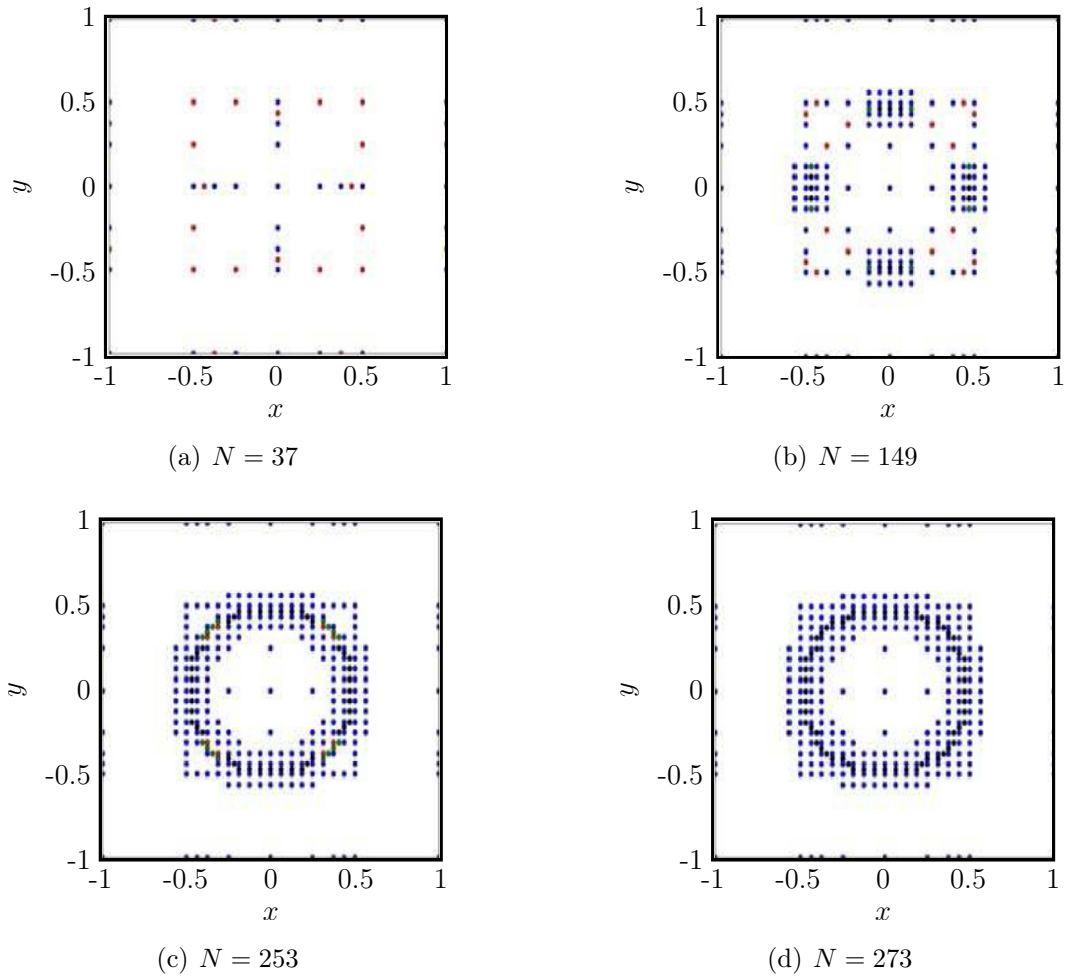
where $S^{[\ell]}(b)$ is the set of $m+1$ closest points around $y^{[\ell](b)}$ in the $y^{(\ell)}$ directions. The coefficients $c_j(y^{(\ell)})$ are obtained with polynomial annihilation in ℓ directions by solving a linear system. The denominator $q_m(y^{(\ell)})$ is computed as in Equation (2.14).

With this new stochastic approach, the authors have introduced an adaptive domain classification. A general polynomial chaos expansion is applied on the initial grid. The polynomial annihilation edge detection method combined with the min-mod function is then applied to the polynomial chaos (PC) expansion to detect the zones of discontinuity, classifying the nodes depending on which side of the jump they are. The zones where the neighbor points have different labels are then further refined, and new PC expansions are constructed in order to repeat the procedure.

Jakeman *et al.* have extended this procedure in dimensions larger than 2 [40]. The process of adaptive refinement applied to a 2D circular discontinuity is illustrated in Figure 2.6, with N the number of evaluated points. As seen, the number of points away from the discontinuity is significantly reduced compared to the initial method of Archibald without spectral expansion of the jump function u. Nonetheless, the number of points in the proximity of the jump is huge when a precise description of the curve is necessary. Indeed, the polynomial chaos expansion at the fine grid close to the discontinuity may not be satisfactory due to the Gibbs phenomenon. Therefore, the method relies on the evaluation of the function to increase the accuracy of the discontinuity location. In addition, their domain classification method relies on a tensor grid, which is not sufficiently general.

When using polynomial annihilation approach, the discontinuity localization is exact, which was not the case with the Bayesian approach. However, the main disadvantage of polynomial annihilation is the number of evaluations which can become large due to the fine tensor grid nearby the discontinuity. It is, therefore, a good practice to preliminary locate and estimate the size of the discontinuity locations before applying a local refinement with another method. Different authors have proposed methods that use polynomial annihilation as a preliminary step to detect the position of the discontinuity broadly and use other approaches to refine the mesh and increase the accuracy of the detection. Gorodetsky *et al.* [33] and V. Halder *et al.* [36] have proposed two discontinuities localization methods based on this approach. Both authors use another useful machine learning tool named Support Vector Machine. Therefore, before describing the discontinuity detection approaches, the Support

Vector Machine is introduced.



(a) $N = 37$ (b) $N = 149$

(c) $N = 253$ (d) $N = 273$

Figure 2.6: Adaptive refinement method using polynomial approximation of the jump function [40].

2.3.4 Support Vector Machine

Support Vector Machine is a common machine learning tool that is used to construct a classifier to predict which side of the classifier, namely which class, a point belongs to, given a set of N training data. The N input points are labeled depending on each class they are. From this knowledge, Support Vector Machine determines one or multiple hyper-plane that separates the domain into different classes, given these labeled points. The classifier is then used to predict which class a given point of the domain belongs to.

For example, if a linear model separating two classes can be constructed, the SVM tries to place a linear boundary between the negative (■) and the positive (○) classes, as illustrated in Figure 2.7. The boundary is oriented in such a way that the margin (the area between (- - - -)) is maximized. Choosing the best solution among the multiple solutions classifying

exactly the training data set maximizes the distance between the boundary (——) and the nearest data point (■), called the Support Vectors [13].

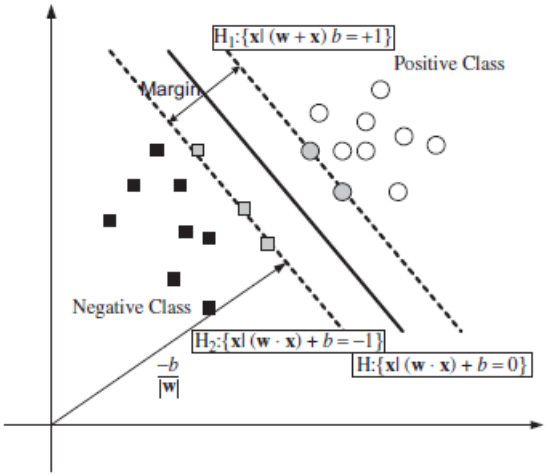


Figure 2.7: Linear Support Vector Machine classifier [73].

More generally, the classifier is defined by a function such as

$$f(\mathbf{x}) = \mathbf{w}^T \Phi(\mathbf{x}) + b, \tag{2.18}$$

where $\mathbf{w}$ is the normal vector to the hyperplane, $\Phi(\mathbf{x})$ is a fixed feature space transformation and b is an explicit biased parameter [13]. The classifier takes the value

$$f(x_i) = 1 \text{ if } y_i = 1 \qquad \text{or} \qquad f(x_i) = -1 \text{ if } y_i = -1 \quad \text{with } i = 1, \ldots, N, \tag{2.19}$$

where y_i is the label of the points x_i, corresponding to the class it belongs to. Points of the positive class are labeled 1, while negative class points are labeled -1. The values of $y_i f(x_i)$ is thus always positive, except at the location of the hyperplane, where $f(x_i) = 0$. The value $f(x_i) \approx 0$ corresponds to the zone where the classifier has the lowest reliability, meaning that the points of this zone can not be classified. These equations can be rewritten as

$$y_i f(x_i) = y_i(\mathbf{w}^T \Phi(x_i) + b) \geq 1 \qquad i = 1, \ldots, N. \tag{2.20}$$

The margin is maximized and the points that lie on the wrong margin boundary are penalized. Mathematically, the optimization problem writes

$$\begin{aligned} &\min \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{M} \zeta_i \\ &\text{subjected to} \\ &y_i(\mathbf{w}^T \Phi(x_i) + b) \geq 1 - \zeta_i, \qquad \zeta_i \geq 0, \qquad i = 1, \ldots, N. \end{aligned} \tag{2.21}$$

Indeed, maximizing the margin which is proportional to $1/\mathbf{w}^2$, corresponds to minimizing $\mathbf{w}$. In practice, since the predictions are not perfect, we allow some points to be at a distance ζ_i from their correct margin boundary. This acceptable margin is controllable with the penalty term C. This parameter is the inverse of a regularization coefficient. It controls the trade-off between minimizing training errors and controlling model complexity. The error margin is minimized for a large C.

For linear separation, the geometrical margin corresponds to $\|\mathbf{w}\|^{-2}$ and is maximized to select the hyperplane as illustrated in Figure 2.7. For nonlinear classification tasks, Kernel functions are used. The inputs points are mapped onto a high-dimensional feature space, where a linear classification is possible. Φ is the nonlinear vector function that is used to map the n-dimensional input vector $\mathbf{x}$ onto n-dimensional feature space. In order to avoid computational problems due to large vector or overfitting, Kernel functions are introduced, returning a dot product of the feature space mapping $K(\mathbf{x_i}, \mathbf{x_j}) = \Phi(\mathbf{x_i})^T \cdot \Phi(\mathbf{x_j})$. Each function that satisfied Mercer's theorem can be used as Kernel function [73, 67]. The main Kernel functions used for SVM's are

- polynomial homogeneous: $K(\mathbf{x_i}, \mathbf{x_j}) = (\mathbf{x_i} \cdot \mathbf{x_j})^d$, with d the degree
- polynomial inhomogeneous: $K(\mathbf{x_i}, \mathbf{x_j}) = (\mathbf{x_i} \cdot \mathbf{x_j} + 1)^d$, with d the degree
- Gaussian radial basis function (rbf): $K(\mathbf{x_i}, \mathbf{x_j}) = \exp(-\gamma\|\mathbf{x_i} - \mathbf{x_j}\|^2)$, with $\gamma > 0$
- hyperbolic tangent: $K(\mathbf{x_i}, \mathbf{x_j}) = \tanh(\kappa\mathbf{x}_i \cdot \mathbf{x}_j + c)$, with $\kappa > 0$ and $c < 0$

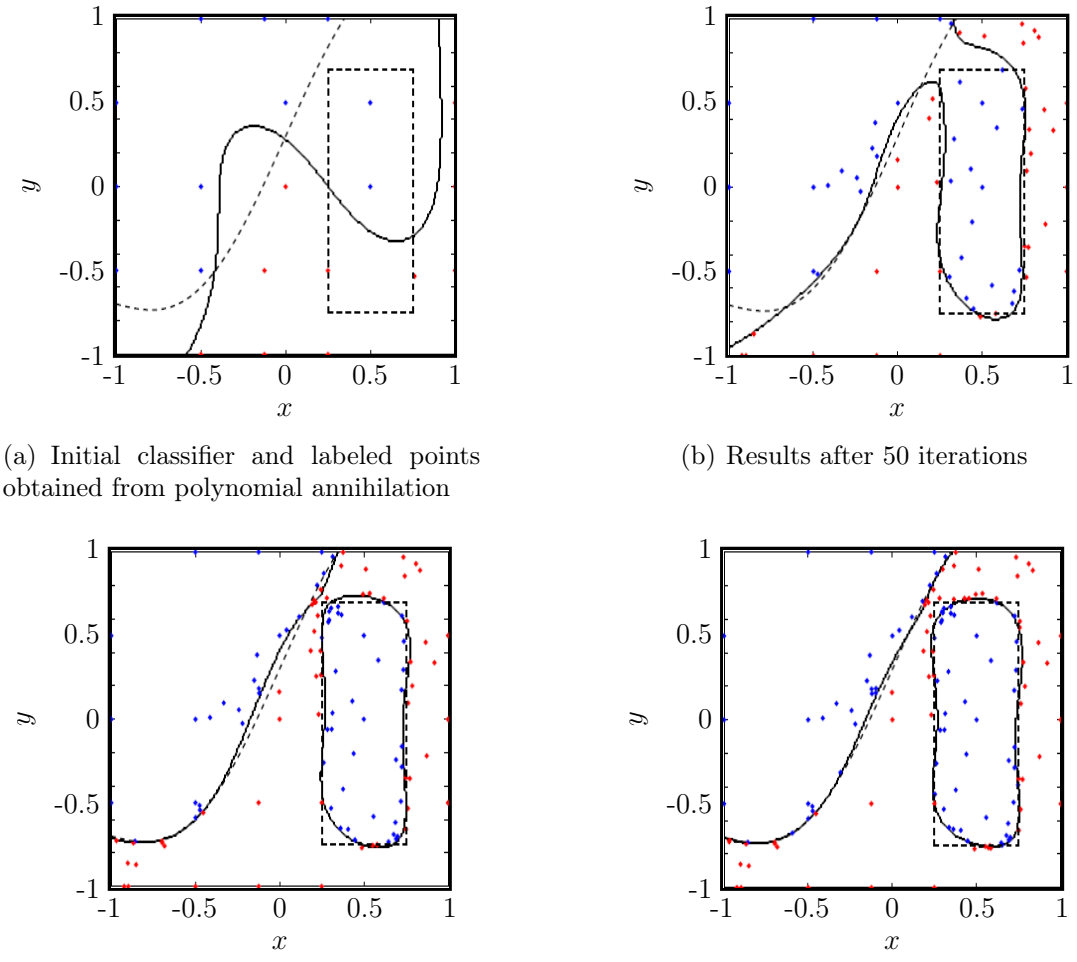
In particular, for the Gaussian rbf Kernel, γ indicates how far a single training point influences the model, it corresponds to the inverse of the radius of influence of a given point. Large value of gamma leads to a fine resolution of the classification boundary but can result in overfitting and no C can regularize the error. The score of the SVM characterizes the likelihood that a label comes from a particular class, or in other words, the probability that a point is assigned to the correct class.

Providing this theoretical background, let's focus on the description of the discontinuity detection methods relying on polynomial annihilation and Support Vector Machine.

2.3.5 Methods using polynomial annihilation and SVM

The first application of polynomial annihilation combined with another method to locally refine the domain was introduced by Gorodetsky and Marzouk [33]. They use 1D polynomial annihilation functions in various directions j, to initiate the domain decomposition and generate the first set of labeled points. The jump functions are evaluated in j direction with $j = 1, ..n$ where n is the dimension of the space.

The initial set of evaluated points (see Figure 2.8(a)) is used to determine the jump function values at various positions of the parameter space. The initial set of points $\mathcal{S}_x$ is constructed based on an off-axis tolerance and nearest neighbor approach. Suppose a jump is detected at a node $\mathbf{x}^k$, in direction j, the model $u(\mathbf{x}^k)$ is evaluated at this location. The point $\mathbf{x}^k$ is added to the set of nodes, and the edge detection method is applied in another direction. The same procedure is performed iteratively until the maximum number of evaluated points has been reached, or no further refinement is possible. The points are labeled according to the set of evaluated points. A circle of radius δ is considered around $\mathbf{x}^k$ and the point inside the circle which has the maximum response $u(\mathbf{x}^*)$ is assigned to class 1. Then the evaluations of all other points $u(\mathbf{x})$ in the given circle are compared to this value $u(\mathbf{x}^*)$. If the difference between $u(\mathbf{x}^*)$ and $u(\mathbf{x})$ is below the value of the jump function value, the nodes is labeled -1; otherwise, it is assigned to class 1.



(a) Initial classifier and labeled points obtained from polynomial annihilation

(b) Results after 50 iterations

(c) Results after 100 iterations

(d) Results after 120 iterations

Figure 2.8: Labeled points: class 1 (•) and class -1 (•) and SVM classifier (——) obtained moving forward in the process [33].

Given the set of labeled points, Support Vector Machine classification is used to find the

discontinuity location. A Gaussian kernel is chosen by the authors [33]. Once the preliminary location of the discontinuity is known, the algorithm adds points around this zone to increase the accuracy of the detection. This approach advantageously concentrates computational efforts near the discontinuity and avoids distributing points where there is no discontinuity. Points are added randomly along the discontinuity based on their proximity to the zero-level set of the classifying surface. The labeling process still relies on the jump function value. Each time a point is evaluated and labeled, the Support Vector classifier procedure is repeated. The stopping criterion relies on a minimal distance between two points of the domain. The process is illustrated in Figure 2.8. The exact jump location (- - - -) and zero-level of the Support Vector classifier (——) obtained are represented in Figure 2.8. The local refinement process allows improving the discontinuity detection. The method of Gorodetsky *et al.* [33] allows the detection of multiple discontinuities and does not require any geometrical prior knowledge. Nonetheless, as visible in Figure 2.8, SVM does not allow to accurately detect sharp edge discontinuities.

V. Halder *et al.* [36] have introduced another detection algorithm using the polynomial annihilation method and Support Vector Machine to overcome a limitation of the edge tracking methods previously described. All the techniques described previously rely on the detection of discontinuity and did not apply in the case of smooth quantities of interest. However, without any knowledge of the quantity of interest, the methods can be useless since they will search for discontinuities even if the model is smooth. To overcome this limitation, the authors have developed an adaptive method based on the local gradients and probability density function (PDF) of the quantity of interest by using *Minimum Spanning Trees Multi-Element* [36].

In graph theory, minimum spanning trees corresponds to a graph that connects all the vertex together by minimizing the sum of the weight of all the edges. In the paper, the weights of the edges w_i are based on the gradient, or on the PDF of the model of interest u, or even on a combination of both. Long edges with large gradients or/and edges with a high PDF value correspond to a discontinuity region. The edge weights are used to identify the most relevant location to add nodes. Edges with weights below a given threshold $w_i < w_{min}$ are refined. A point is added in the middle of these edges. The process stops when the maximum number of evaluated points or iterations is reached. Once the stopping criterion is satisfied, the domain is divided into two subdomains using Support Vector classification. First, edges crossing a discontinuity are determined using the values of the gradients of the edges. The largest gradient edges are considered as crossing a discontinuity. The labeling process is the same as Gorodetsky [33], using the jump value of the polynomial annihilation edge detection. The SVM is used using c different classes depending on the number of discontinuities present in the domain, and the Gaussian rbf Kernel is fixed. The performances of the method are

studied in the context of shallow water dam breaks. Gorodetsky *et al.* have then discussed the performance of this method [74]. The main disadvantage of this approach is that there is no distinction between zones of discontinuities and zone with a steep slope.

The significant improvement introduced thanks to the combination of polynomial annihilation and an isolated method of local refinement [33, 36], compared to the adaptive refinement process of Archibald *et al.* [7] and Jakeman *et al.* [41] is illustrated in Table 2.1. The number of evaluated points is indeed reduced by a factor 100.

	Gorodetsky & V. Halder [33, 36]	Archibald [7]	Jekeman [41]
model evals for 1% error	~ 130	$31,376$	$91,250$

Table 2.1: Performance comparison between the different methods of discontinuity localization found in the literature [33].

2.4 Summary

Local approximation methods based on adaptive grids are useful for high-dimensional problems since they take advantage of the exponential convergence of spectral methods applied to the smooth subdomains. Methods based on both gPE and multi-wavelets have shown good results for discontinuous function approximations. Nonetheless, the construction of a reliable approximation model on each subdomain requires a dense space grid on each domain, increasing computation times drastically. The global approximation has been introduced to reduce the computational expense of the local approach but does not provide a precise approximation of the model in the vicinity of the jump.

Domain decomposition techniques based on edge tracking detect discontinuities more efficiently and separate the problems of discontinuity detection and surrogate construction. However, current methods do not deal with discontinuities that run partially through the domain, which can be characteristics of nonlinear response in the context of blade/casing interactions. The method proposed in this works relies on the edge tracking approach and aims to provide improvement compared to the methods already implemented. The proposed methodology to localize discontinuities in 2D Design of Experiment is presented in the next chapter, constituting the main core of this work.

3 | Proposed algorithm

Based on the theoretical background presented in chapter 2, an algorithm to localize discontinuities in nonlinear Design of Experiment inputs space has been developed. This chapter is divided into three main sections. The first part aims to describe the overall methodology followed and the main steps of the numerical method implemented. Multiple strategies were tested, and the final choices are discussed and justified, using an example to illustrate key stages. In the second part, numerical procedures introduced to increase the algorithm efficiency and robustness are described. The last section highlights the sensitivity of the proposed numerical method to some of its numerical parameters.

3.1 Overview of the proposed methodology

The algorithm was implemented in *Python 3.8*. The method has to be as general as possible to enable the localization of a wide variety of discontinuities. In the specific context of nonlinear responses, a meaningful solution must feature discontinuities that run partially through the domain and deal with models containing zones with steep slopes. The followed strategy is described in the flowchart of Figure 3.1. The numerical model of interest $u(\mathbf{x})$, is given by the user, where $\mathbf{x} = (x, y)$ since the method is strictly limited to two dimensional parameters space. The evaluated function is deterministic so that the evaluation of the model at a given point is the same when repeating the evaluation. The model must be evaluated at the most relevant location to limit computation times. To guarantee the termination of the process, the user specifies the maximum number of points N_{max} evaluated during the whole process. The outputs are the control points $\mathbf{Q}_x$ and $\mathbf{Q}_y$ of each B-spline describing the discontinuity locations in the input space (x, y).

The proposed approach takes advantage of an anisotropic Delaunay triangulation of the domain [26] to evaluate the zone of discontinuities using triangle characteristics. The model is evaluated on an basic set of points and a first Delaunay mesh is constructed (Block A). Polynomial annihilation method [8] and computation of gradients are combined to construct the detection tool and a global refinement guided by these mesh features are conducted until the zones of discontinuities are sufficiently refined. The points added to the domain nearby the discontinuity are labeled on either side of the discontinuity based on the region they belong to (Block B). The triangles containing the same discontinuity are clustered using neighbors characteristics (Block C). These labeled points belonging to the same discontinuity triangles are then exploited to train a Support Vector Machine model [16] that creates

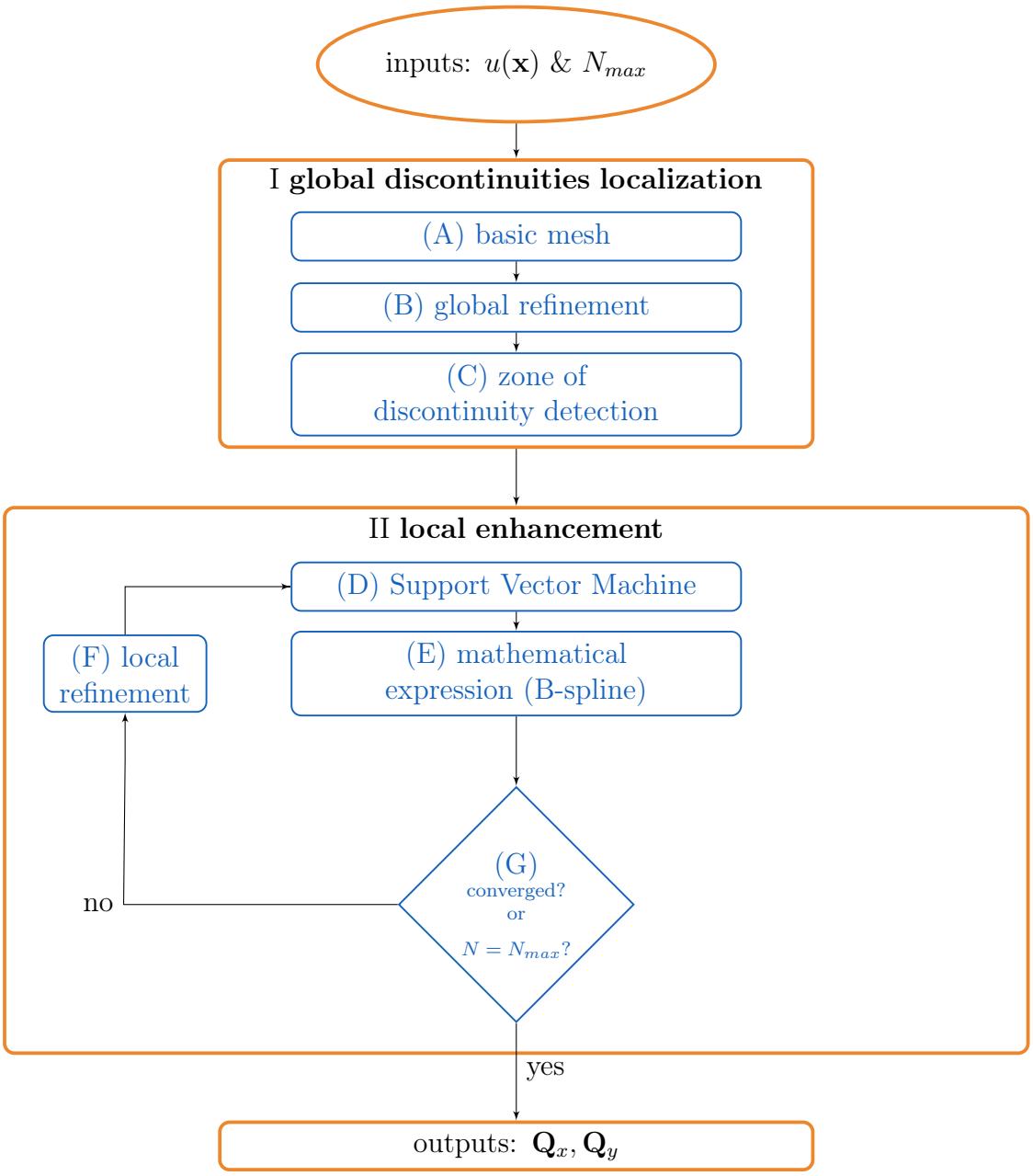


Figure 3.1: Proposed algorithm.

a smooth approximation of the boundary between the domains, giving the discontinuity location (Block D). The separation line is described mathematically by means of a B-spline, defined by its control points (Block E). Finally, local refinement in the vicinity of the discontinuity is performed using a priority indicator (Block F). Sampling stops when the maximum number of points N_{max} specified by the user is achieved or when the solution has converged to a sufficiently stable discontinuity location. Tow criteria are used together to asses the convergence of the gathered B-spline (Block G).

The model $u(\mathbf{x})$ used to illustrate the algorithm description is the Duffing oscillator, a

nonlinear second-order differential equation, which writes

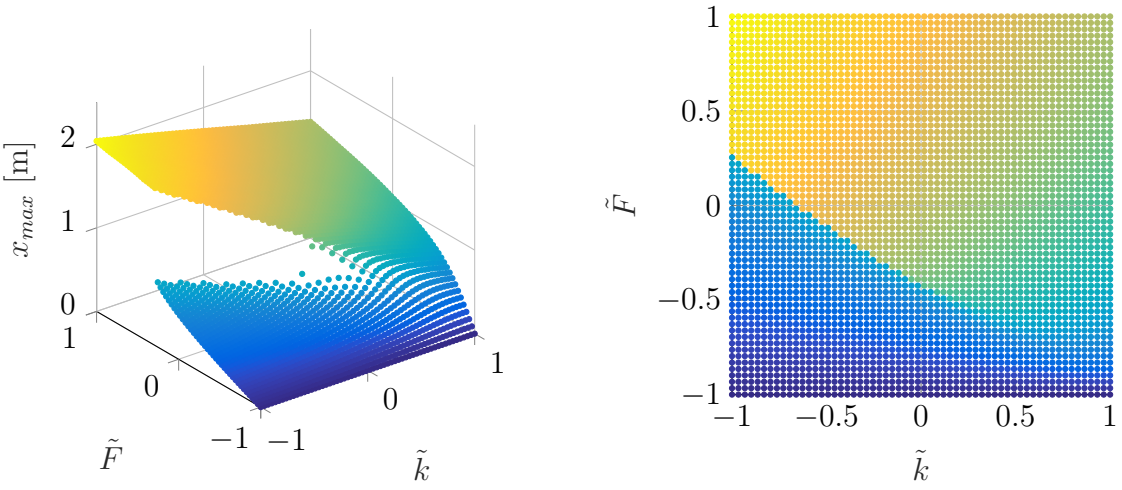
$$m\ddot{x} + c\dot{x} + kx + k_{nl}x^3 = F\cos(\omega t), \tag{3.1}$$

where m is the oscillator mass [kg], c is the damping coefficient [kg/s], k is the linear stiffness [N/m], k_{nl} is the cubic stiffness, F is the amplitude of the force [N] applied to the mass, ω is the pulsation [rad] and t is the time [s]. The variables studied are the linear stiffness k and the amplitude of the force F, varying from 0 to 1. The output of interest is the maximum displacement of the mass x_{max} [m] on a period T. The model coefficients values are summarized in Table 3.1.

m	c	$k_n l$	k	F	ω	T
1 kg	0.25 kg/s	0.5 N/m^3	[0;1] N/m	[0;1] N	1 rad/s	126 s

Table 3.1: Duffing oscillator parameters.

The input values are normalized so that $\tilde{k} \in [-1;1] \times \tilde{F} \in [-1;1]$. The maximum displacement of the mass x_{max} as a function of the two normalized parameters $\tilde{c}$ and $\tilde{k}$ is illustrated in Figure 3.2. A jump is visible in the response. The latter is due to the nonlinear behavior introduced by the cubic spring. This example has the advantage of illustrating the detection of partially crossing discontinuity with zones of large slopes, which are two challenges of this work.



(a) 3D view

(b) 2D view

Figure 3.2: Reference solution of the Duffing oscillator.

The proposed methodology introduces innovations and advantages compared to those found in the literature [33, 36]. One innovation is the combination of polynomial annihilation

and the computation of gradient that allows the use of small degree polynomial annihilation to distinguish zones of discontinuity and steep slope. Besides, the proposed method takes advantage of a Delaunay triangulation to localize the zone of discontinuity. Unlike the Minimum Spanning Trees Multi-Element methods of V. Halder [36], and the approached of Gorodetsky [33], the suggested methodology allows the detection of discontinuities that run partially through the domain. Finally, the stability criteria are good indicators of convergence that have never been used in the past. The two next sections describe the two main shares, block **I** and block **II**, of the proposed algorithm, following the methodology summarized in the flowchart of Figure 3.1.

3.2 Global discontinuities localization (I)

The first part of the algorithm consists in a first estimation of the discontinuities' locations and sizes. First, a basic mesh is constructed. During the global refinement, nodes are added where large gradients and jump functions occur, and each new point is labeled depending on which side of the discontinuity it is located. This section details each of these steps. Ideally, the global refinement would be coarse in smooth areas and fine in the zones of discontinuities.

3.2.1 Basic mesh (Block A)

The initial nodes combine points of a regular grid ($n^0 \times n^0$ points) and n_{LHS} points distributed in the whole domain with a *Latin hypercube sampling* to optimally cover the entire domain [38], as illustrated in Figure 3.3. The regular grid (•) guarantees that the whole domain is covered since there is, at least, a point at the four corners of the domain. Latin hypercube nodes (•) allow removing the regularity of the isotropic mesh and help reaching the convergence as discussed in subsection 3.5.1.

The initial number of points has to be carefully chosen. If too small, the risk is to be unable to detect zone of discontinuities. For example, in a typical nonlinear response, the value of two points located at a far distance from a jump could be the same and the lack of node in the intermediate zone would result in failing to detect the jump. In the considered example, the size of the regular grid is fixed to $n^0 = 6$, and $n_{LHS} = 10$ randomly distributed nodes are added. The influence of the initial nodes number N_{basic} on the discontinuity detection is discussed in subsection 3.5.1. If no zone of discontinuity is detected with this basic mesh, a warning is sent advising the user to add points. This may happen if the discontinuity covers a small portion of the domain. Nonetheless, the method described in this report is not well suited for the detection of localized discontinuities covering only a small part of the domain. Other techniques have been developed for such problem, especially in the context of crack propagation [24].

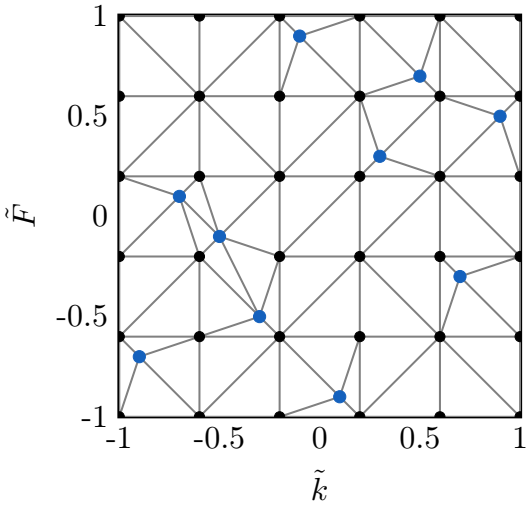


Figure 3.3: Basic mesh configuration (regular grid n^0 (•), Latin hypercube nodes n_{LHS}(•)) and the corresponding Delaunay triangulation (——).

Then, a basic Delaunay triangulation is constructed based on the initial nodes, as shown in Figure 3.3 (——). By definition, *a triangulation of a set of points A is a triangulation of Delaunay if, for any triangle t of the triangulation, no point of A is strictly inside the circumscribed circle at the triangle t* [42]. Delaunay triangulation characteristics allow to avoid very obtuse or acute angle in any triangle and ensure a certain regularity in the mesh. Delaunay triangulation usually goes hand in hand with the Voronoi domain decomposition [36]. A Voronoi cell corresponds to the region for which each point is closer to one given vertex of the Delaunay triangulation than another, as illustrated in Figure 3.4. For example,

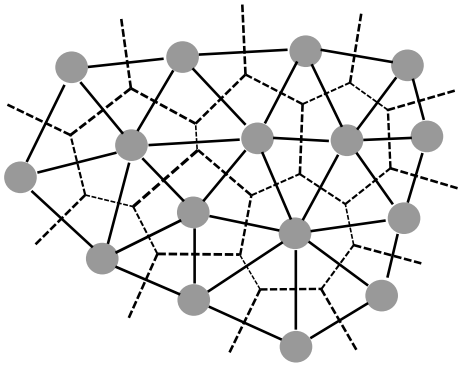


Figure 3.4: Voronoi (- - - -) and Delaunay (——) meshes for a given set of points [20].

Rushdi *et al.* [60] has used the Voronoi cells to define the zones of discontinuity rather than the Delaunay triangles. He uses the neighbor information of each cell . Similarly, Debnath uses Voronoi cells and neighbors characteristics such as the number of neighbors, the area of the neighboring cells to define the different zones of continuities [21]. However, this use

of Voronoi cells is poorly suited for the algorithm developed in this work since it required too much evaluation of the model. In addition, a Delaunay triangulation is better suited than a Voronoi for randomly distributed data [26, 27]. Finally, the anisotropic feature of a Delaunay triangulation will be useful since new nodes will change the orientation of different edges possibly detecting other zones of discontinuity without additional numerical cost.

Once the basic Delaunay mesh is constructed, as illustrated in Figure 3.3, the global refinement process can begin.

3.2.2 Global refinement (Block B)

A good global mesh would be coarse in smooth areas and fine in the zones of discontinuities. Points must be added in the most informative places to limit computation times. For that purpose, an adaptive refinement technique based on the triangle characteristics is implemented.

Triangle characteristics

The refinement criterion relies on two indicators: (1) the gradients of the triangle and (2) the jump function amplitude of the polynomial annihilation edge detection of Archibald [8]. These characteristics are computed for each triangle and are used to detect the zones of discontinuity.

The quantity $\dfrac{\partial u^i(\mathbf{x})}{\partial \mathbf{x}^i}$ is an estimation of the local variation of the field $u(\mathbf{x})$ on edge i. By definition, zones of discontinuities correspond to zones where the gradient is large. For that purpose, the edge gradients between two vertices of a Delaunay triangle are computed using the coordinates (x, y) of the nodes and the corresponding response $z = u(x, y)$. For a convex polygon, such as a triangle, the unit normal vector $\boldsymbol{n}$ at the middle point, illustrated in Figure 3.5, can be computed using the cross product of two edges of the polygon. Any three points $\boldsymbol{P_0}, \boldsymbol{P_1}, \boldsymbol{P_2}$, with $\boldsymbol{P_i} = (x_i, y_i, z_i)$, determine a unique plane. To find the normal to this plane, the first step consists in constructing the vectors of difference of each coordinate such as

$$\begin{aligned} \mathbf{D}_1 &= \boldsymbol{P_0} - \boldsymbol{P_1} = (x_0 - x_1, y_0 - y_1, z_0 - z_1) \\ \mathbf{D}_2 &= \boldsymbol{P_0} - \boldsymbol{P_2} = (x_0 - x_2, y_0 - y_2, z_0 - z_2)\,, \end{aligned} \tag{3.2}$$

where $\boldsymbol{P_0}, \boldsymbol{P_1}, \boldsymbol{P_2}$ are the three vertices of the considered triangle. The normal vector, illustrated in Figure 3.5, is then given using the cross product of these two vectors such as

$$\mathbf{n} = \mathbf{D}_1 \wedge \mathbf{D}_2 = (n_x, n_y, n_z)\,. \tag{3.3}$$

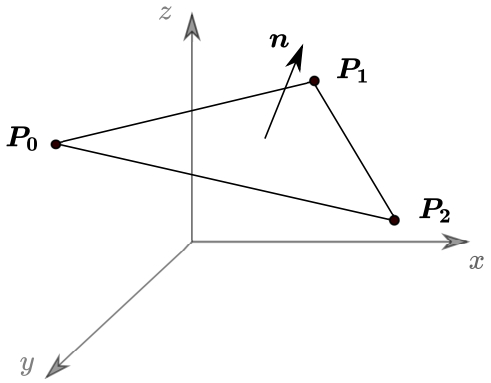


Figure 3.5: Normal vector $\boldsymbol{n}$ of one triangle.

The gradients along x and y are then computed such as

$$\text{grad}_{\mathbf{x},\mathbf{y}} = \left(\frac{n_x}{n_z}, \frac{n_y}{n_z} \right). \tag{3.4}$$

Finally, in order to have a unique scalar field per triangle, the norm of the gradients is computed using

$$\| \text{grad}_{\triangle} \| = \sqrt{\text{grad}_{\text{x}}{}^2 + \text{grad}_{\text{y}}{}^2}. \tag{3.5}$$

The gradients of each triangle for the basic mesh, in the case of the Duffing oscillator, are illustrated in Figure 3.6(a). As expected, the largest amplitude appears where the jump occurs in the response, as seen in Figure 3.2.

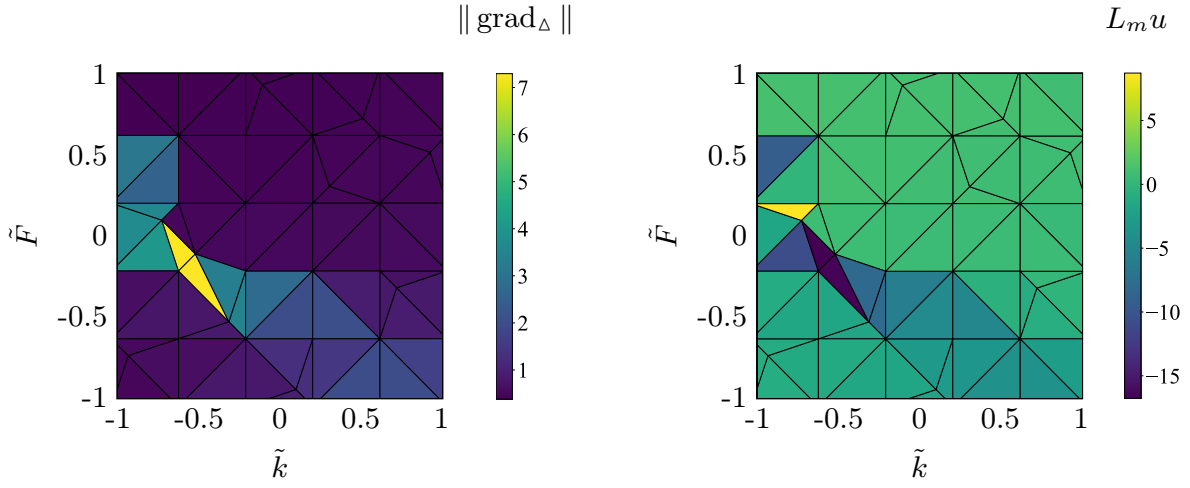
However, this gradient based indicator is not always efficient, as there is no distinction between a discontinuity and a steep slope area. Therefore, a second criterion is used based on the polynomial annihilation method. As detailed in subsection 2.3.2, the method identifies jump discontinuity at $\mathbf{x}$ using enclosed points $S_{\mathbf{x}}$ [8]. In that case, the jump function is computed at the centers of the triangles, and the enclosed points are the Delaunay triangle vertices. A jump function $L_m u(\mathbf{x})$ of degree two ($m = 2$) is constructed at each triangle center, by using the three vertices ($m + 1$ enclosed points) [8], such as

$$L_m u(\mathbf{x}) = \frac{1}{q_{m,2}(\mathbf{x})} \sum_{\mathbf{x}_j \in \mathcal{S}_{\mathbf{x}}} c_j(\mathbf{x}) u\left(\mathbf{x}_j\right). \tag{3.6}$$

The coefficients c_j are computed solving a linear system and the denominator $q_{m,2}$ is obtained by summing the c_j coefficients. More details are given in subsection 2.3.2. By definition, the value of the jump function is large for zones of discontinuities but not for zones of large gradients.

The low degree m of the approximation functions $L_m u$ limits the rate of convergence [8]. Nonetheless, the anisotropy of the Delaunay triangulation allows the nonuniform distribution

of the enclosed points. This allows the distinction zones of steep slope and zones of large gradients even if the rate of convergence is not optimal. The amplitude of the jump function for each triangle of the basic mesh are illustrated in Figure 3.6(b). Larger values occur at the jump location. The combination of the two indicators, namely the norm of the gradients and the jump function, has shown to give much better results than when evaluated individually.



(a) Norms of the gradients

(b) Jump function values

Figure 3.6: Norms of the gradients $\| \operatorname{grad}_\triangle \|$ and jump function values $L_m u$ for each triangle of the basic mesh.

Refinement criteria

The triangles which are candidates for refinement must satisfy two requirements. First, a minimal distance ε between two nodes of the mesh is fixed. If a new point should be added close to another point at a distance lower than ε, no point is added. The shorter edges triangles are thus discarded. This allows progress in the process and avoids the cluster of nodes in the zone of largest gradients. All the triangles that can not be further refined are no longer considered candidates. As an input, the user specifies the minimal distance ε as a percentage e of the length of the x-axis, such as

$$\varepsilon = e(x_{max} - x_{min}). \tag{3.7}$$

For the purpose of this report, the percentage has been set to $e = 4\%$, corresponding to a minimal distance of $\varepsilon = 0.08$. To visualize the portion of the domain that it represents, a square of edge 0.08 (■) is represented in Figure 3.7. The repercussions of ε on the mesh are discussed later in subsection 3.5.2.

The second condition relies on the value of the jump function $L_m u$ of the remaining candidates. The candidates to refinement are the triangles which jump function values

are outliers, considering only the remaining candidates' jump function values. In statistics, outliers are data points that differ significantly from other observations. Various methods exist to define outliers [63]. In this case, Tukey's fences are used [63], where the upper $L_m^{max}u$ and lower $L_m^{min}u$ limits are defined by using the value of the first $\mathrm{Q}_1^{L_m u}$ and third $\mathrm{Q}_3^{L_m u}$ quartile of the jump function amplitudes such as

$$L_m^{min}u = \mathrm{Q}_1 - 1.5\,(\mathrm{Q}_3 - \mathrm{Q}_1) \qquad \text{and} \qquad L_m^{max}u = \mathrm{Q}_3 + 1.5\,(\mathrm{Q}_3 - \mathrm{Q}_1)\,. \tag{3.8}$$

Triangles that jump function value is above L_m^{max} or below L_m^{min} are considered as outliers and refinement candidates. If Tukey's fence approach does not detect any outliers, the candidates are defined as the triangles that have a jump function value in the largest 10% or the lowest 10%. The candidates to refinement resulting from the first iteration of global refinement are illustrated in Figure 3.7. As expected, they correspond to the largest jump function and norm of norm gradients values (see Figure 3.6(b)).

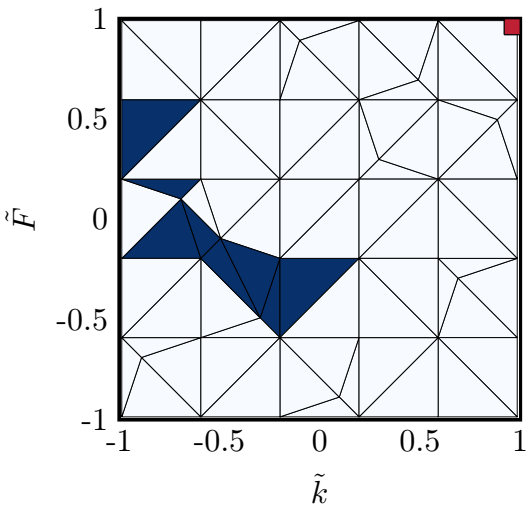


Figure 3.7: Refinement candidates (▲) of the first iteration of the global refinement and a square (■) of edges ε.

Now that the refinement candidates triangles have been selected (▲), each triangle is analyzed and a point is added in the middle of the edge having the vertices with the largest difference in the response. This process is illustrated in Figure 3.8 (left). Then, the system's response is evaluated for each new point and the added node is labeled depending on which side of the discontinuity it is located. The labeling procedure is illustrated in Figure 3.8 (right). The points located in the blue area are considered as 'below' the discontinuity and are labeled -1 and the orange upper region points are classified 1. This classification is based on the value of the evaluation of the function. For a new point with an evaluation z_{new}, the

labeling process is given by

$$\begin{aligned} &\text{if} \quad z_{new} < \frac{|z1 - z2|}{2} \Rightarrow -1; \\ &\text{if} \quad z_{new} >= \frac{|z_1 - z2|}{2} \Rightarrow 1; \end{aligned} \tag{3.9}$$

where 1 and 2 are the triangle vertices (see notations of Figure 3.8). To increase the number of labeled points, the vertices of the edge where the new point is added are labeled if they are not yet, depending on their $u(\boldsymbol{x})$ value.

When a new point is added and labeled, a new triangulation is locally required and the norms of the gradients $\| \operatorname{grad}_{\triangle} \|$ and jump functions $L_m u$ of the new triangulation are recomputed. The process is then repeated to progress in the global refinement. Nonetheless, when new nodes are added, the Delaunay mesh is not modified everywhere [29]. Therefore, the algorithm takes advantage of this property to recompute the characteristics of the new triangles only. More details about this local re-meshing are given in subsection 3.4.1.

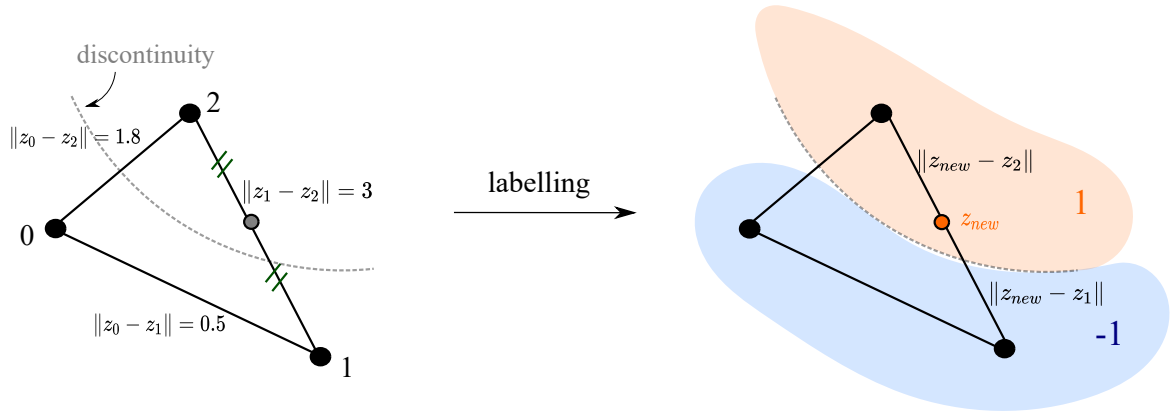


Figure 3.8: Schematic representation of the global refinement and labeling methodologies.

Termination criteria

The maximum number of evaluated points N_{max} is specified by the user to limit computation times. Nonetheless, the algorithm will stop before reaching the maximum number of nodes if the mesh is sufficiently refined in the zones of discontinuity.

Depending on the discontinuity's shape, two conditions must be met to consider that the mesh is sufficiently refined. Firstly, if the jump function runs through the whole domain or is a closed line, the process should stop when there is no more jump to detect, but still zones of large gradients. Therefore, the criterion based on the jump function is applied. If there is no outlier in the value of the jump function of the remaining candidates and if the multiple

of the interquartile (IQR), defined as

$$1.5IQR = 1.5(\mathrm{Q}_3 - \mathrm{Q}_1), \tag{3.10}$$

is below 10%, the process stops.

The second specific case corresponds to a discontinuity that runs partially through the domain. The value of the jump and the gradient will decrease gradually (as seen in Figure 3.2 for the Duffing oscillator). The stopping criterion is thus based on the standard deviation of the norms of the remaining candidates. A normalized measure of the standard deviation of the norm of the gradients $\tilde{s}_n$ is computed such as:

$$\tilde{s_n} = \frac{s_n}{\bar{n}}, \tag{3.11}$$

where n represents the norm of the gradients, s_n is the standard deviation of the norm of the gradients and $\bar{n}$ denotes the mean value of n. The refinement procedure stops if (1) the ratio between the normalized standard deviation of the remaining candidates and the normalized standard deviation of all the triangles is below a threshold and (2) the normalized standard deviation of the remaining candidates itself is below a threshold. The two stopping criteria are summarizes in Table 3.2, c denotes the triangles which are refinement candidates and $\triangle$ all the triangles of the mesh. The given thresholds have been fixed based on different analytical functions to test the robustness of the values chosen.

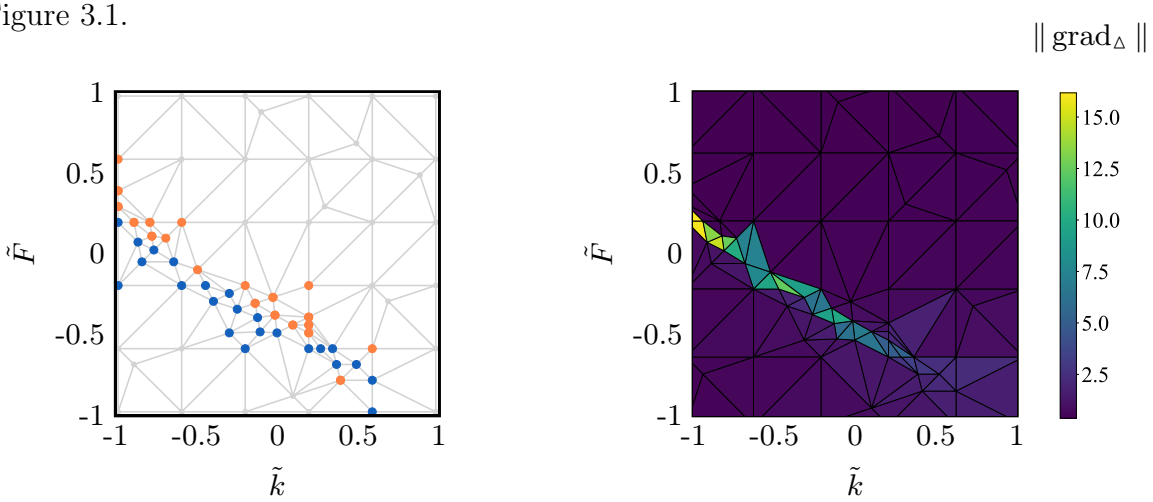
Termination criteria	
case 1	no outlier & $1.5IQR < 0.1$
case 2	$\frac{s_n^c}{s_n^\triangle} < 0.5$ & $s_n^c < 0.6$

Table 3.2: Stopping criteria of the global refinement process.

Global refinement output

The results of the global refinement for the illustrative example are shown in Figure 3.9. As seen, N_{global} evaluated points are added in the vicinity of the discontinuity, and the label describes each side of the discontinuity the node stands (Figure 3.9(a): (•): class -1 and (•) class 1). Compared to Figure 3.6(a), the mesh illustrated in Figure 3.9(b) has been refined in zones where the gradient is high, clearly highlighting the discontinuity location, while the mesh remains coarse where the function is smooth. From this knowledge, the triangles that contain discontinuity can be determined. This is the topic of block C of the flowchart

in Figure 3.1.



(a) N_{global} evaluated nodes: (●) class 1, (●) class -1

(b) Norms of the gradients

Figure 3.9: Final global mesh with $N_{global} = 29$ added points and norm of the gradients of each triangle ($\varepsilon = 0.8$).

3.2.3 Detection of the zones of discontinuities (Block C)

The last part of the prior detection consists in clustering the triangles which surround a discontinuity. The largest gradients triangles (see Figure 3.9(b)) are considered zones of discontinuity. There is not always a sharp distinction in the norm of the gradients, in particular in the case of discontinuities that run partially through the domain since the norm of the gradient will decrease gradually. Therefore, the generalized Extreme Studentized Deviate (ESD) test [59] is used to detect outliers in the gradients set. The maximum number of outliers must be specified to initiate the test. To fix the maximum number of outliers, the candidates that could be further refine are considered and the largest norm of the gradient is retained as a minimal threshold. The number of triangles of the whole mesh, that have a norm of the gradient larger than this threshold are considered as outliers and the ESD test is applied to validate the outliers. Triangles that contains discontinuity (▲) are represented in Figure 3.10.

Once the zones of discontinuities are detected, the second step consists in clustering the triangles that constitute the same discontinuity. This step is crucial in case of multiple discontinuities. The approach relies on the triangle neighbors. Triangles with only one neighbor are considered as the beginning or end of a discontinuity. Then, given a starting triangle, the neighbors are added to the discontinuity until reaching an ending triangle.

When each discontinuity is described by a set of triangles, the vertices of the triangles are retained for the rest of the algorithm. If one vertex has not been labeled during the global

refinement process, the point is labeled using the response of the other vertices of the triangle and their corresponding labels. This way, the number of labeled points to initiate the second step of the algorithm is optimized. The discontinuity triangles (▲) and the labeled nodes (●)(●) retained for the second main part of the algorithm are illustrated in Figure 3.10. The local enhancement part (block **II** of the flowchart in Figure 3.1) uses these labeled nodes and add points to improve the discontinuity localization. The whole process is presented in the following section.

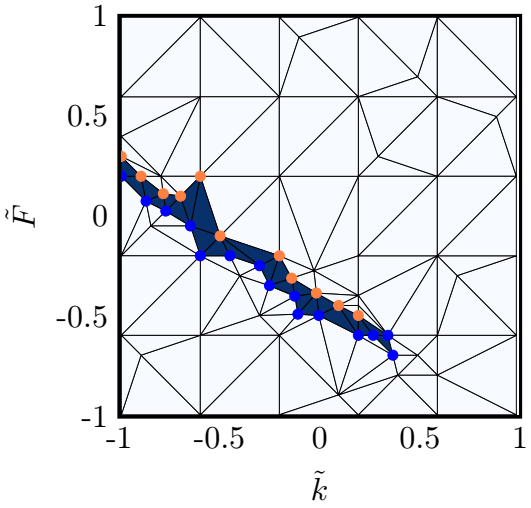


Figure 3.10: Detection of the discontinuity triangles (▲) and labeled nodes: class 1 (●), class -1 (●).

3.3 Local enhancement (II)

The second part of the proposed discontinuity detection tool consists in the localization of the discontinuities based on the prior detection of labeled points during the global refinement. These points are used to construct a SVM classifier. Then, the local refinement process adds new points at the most informative places to improve the accuracy of the discontinuity location.

3.3.1 Support Vector Machine (Block D)

At this stage of the algorithm, the different discontinuities have been localized in a holistic way, using the triangles of the global mesh. The localization of the discontinuity is now deduced from the labeled points distributed in the vicinity of the discontinuity.

A simple polynomial regression could have been considered to find the position of the discontinuity. This approach allows the determination of limits on each side of the

discontinuity, defining a zone containing the discontinuity location. However, this method presents some significant drawbacks that were carefully evaluated. If a high degree is used, the variance will not decrease when increasing the degree due to the Vandermonde matrix that is ill-conditioned [71]. On the opposite, a low degree can limit the precision. Additionally, polynomial regression depends on boundary conditions, causing the curve to oscillate, and providing a poor response if one labeled point is not lined up with the others. Besides, polynomial regression introduces a limitation for the geometry of the discontinuity. For those reasons, the simple polynomial method was discarded and the Support Vector Machine (SVM) [16] was retained as the preferred candidate to localize the discontinuity. Indeed, this method is applicable for many geometries, relatively simple to implement and robust.

As described in subsection 2.3.4, Support Vector Machine is a common machine learning tool used to create a classifier that determines which region a point belongs to. Compared to the k-nearest neighbor approach, another machine learning strategy, SVM allows a more efficient and robust description of the classification boundary [68]. In addition, some parameters allow adapting the expected accuracy of the model depending on the complexity of the classification. As explained in subsection 2.3.4, the margin is controlled by the parameter C and the precision of the model by γ.

The principle of Support Vector Machine classifier is illustrated in Figure 3.11. The SVM model is trained using the labeled nodes: class -1 (•) and class 1 (•). The resulting model is then evaluated on a regular grid illustrated in Figure 3.12. The location of the discontinuity corresponds to the zero-level of the classifier (——), namely the points that can not be classified. These points are used in the next step of the algorithm to find a mathematical expression of the discontinuity location, they are the knot points $\mathbf{P}$ of the B-spline.

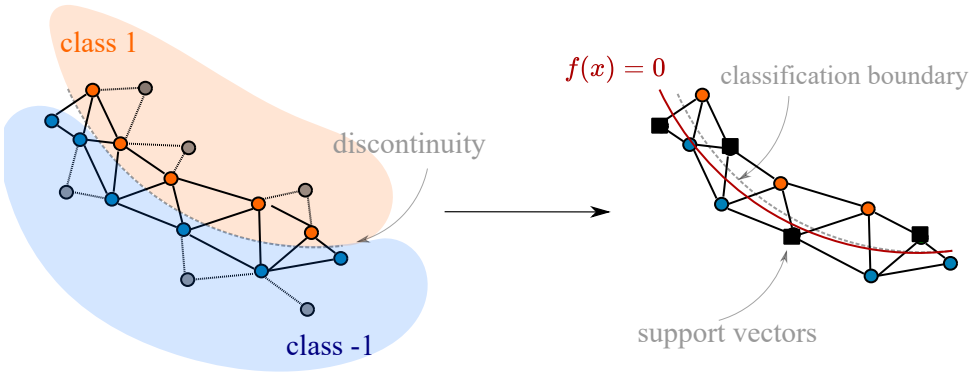


Figure 3.11: Schematic representation of domain decomposition using Support Vector Machine.

The optimal parameters will vary depending on the model studied [14]. A possibility would be to fix a Gaussian rbf Kernel with general values for C and γ, as Gorodetsky et

al. and V. Halder et al. have done in their algorithms [33, 36]. Nonetheless, there is a risk of overfitting, meaning that the model constructed corresponds too closely or exactly to a particular data set and lead to the risk of miss predicting future observations. Indeed, searching for a model more complex than the discontinuity itself can lead to a bad localization of the discontinuity [14]. In the context of blade/casing interaction, the discontinuity is expected to have a polynomial form, and the polynomial kernel will allow converging must faster than the Gaussian kernel in that case, as discussed later. For that reason, the algorithm evaluates the best model (kernel function, C and γ) for each discontinuity. The different possibilities are listed in Table 3.3.

Kernel functions	polynomial ($d = 3$)	rbf	
C	1	10	100
γ	0.01	0.1	1 | 10

Table 3.3: SVM parameters.

The kernel function and the parameters of the method are thus determined thanks to the functions *GridSearch* and *best_ estimator_* implemented in the *scikit learn* package of *Python* [2]. When points are added to the data, the parameters of the model can be reevaluated or fixed depending on the convergence of the parameters at the previous iterations. If the parameters stabilize after adding some points and the score of the model is above 0.9, the parameters are fixed for the rest of the refinement. The score of the SVM gives an indication of the number of points that are well classified. *K-Fold cross-validation* [65] could have been used, but in that specific application, the number of training points is, most of the time, too small to take advantage of the procedure. Further details about the SVM parameters are given in subsection 3.5.3.

The classification boundary (——) obtained in the Duffing oscillator case is illustrated in Figure 3.12. The first approximation of the discontinuity location is quite good referring to Figure 3.2. Nonetheless, some labeled points are badly classified, leaving a place for improvement of the SVM classifier. The evaluation grid of the SVM classifier is also illustrated (.)This grid is bounded by the larger and the lowest coordinates of the labeled points. For the purpose of this report, the size of the grid is fixed so that the whole x-axis is divided into 101 points. The distance between two points of the grid is thus equal to 0.02. Only a small set of labeled points are used as Support Vectors (■). The location of the discontinuity is represented by the points $\mathbf{P}$ (●) lying on the zero-level classifier. These points are used to find the mathematical expression of the discontinuity location, as explained in the following section.

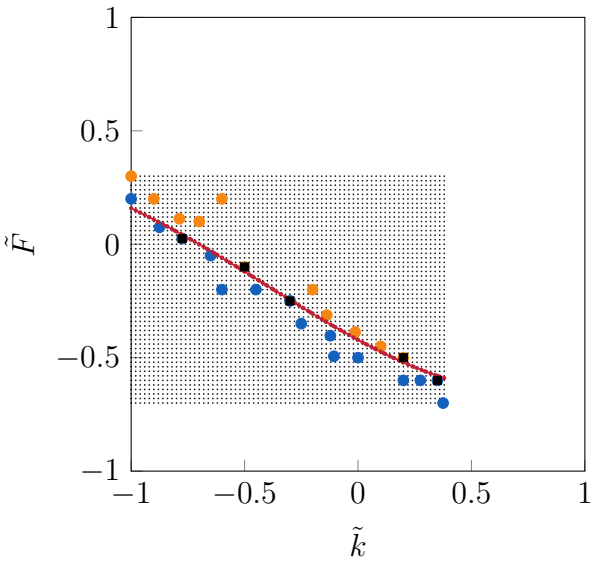


Figure 3.12: Support vector machine applied to the Duffing oscillator: evaluation grid (.), classifier boundary (——), training points: (●) class 1, (●) class -1 and support vectors (■).

3.3.2 Mathematical expression of the discontinuity (Block E)

The localization of the discontinuity must take the form of a simple mathematical expression to, subsequently, subdivide the domain into smooth subdomains. The localization of a discontinuity is thus described with a *Basic-spline* (or B-spline), which is an adequate option. Indeed, the Bézier representation has the disadvantage that the degree of the polynomial is directly linked to the number of control points [56], but a large degree is not practical. Additionally, the modification of one point of the curve changes the description of the entire curve [56], which complicates the local refinement of the discontinuity location. Polynomial regression also has many disadvantages, already detailed in subsection 3.3.1. B-spline [56] has been introduced to remedy these issues.

A B-spline is a piecewise polynomial representation of a smooth curve. The main notations used are illustrated in Figure 3.13. A polynomial of degree d, fixed by the user, is defined on each patch $[p_i, p_{i+1}]$, where p_i is a knot. The coordinates $\mathbf{c}(t)$ of the spline curve write as a function of an intrinsic parameter $t \in [0, 1]$, which follows the curve of the patch. The patch is built as a linear combination of basis function which influence are driven by the control points such as

$$\mathbf{c}(t) = \sum_{i}^{i+d} \mathbf{Q}_i B_{i,d}(t), \tag{3.12}$$

where $\mathbf{Q}_i$ are the control points (●) and $B_{i,d}(t)$ are the *d-degree B-spline* function. The basis

function $B_{i,d}(t)$ is defined on a patch, by using the *Cox-de-Boor* algorithm [5, 34], such as

$$\begin{aligned} B_{i,0}(t) &= \begin{cases} 1 \text{ if } t \in [t_i, t_{i+1}] \\ 0 \text{ if } t \notin [t_i, t_{i+1}] \end{cases} \\ B_{i,d}(t) &= \frac{t-t_i}{t_{i+d}-t_i} B_{i,d-1}(t) + \frac{t_{i+d+1}-t}{t_{i+d+1}-t_{i+1}} B_{i+1,d-1}(t), \end{aligned} \tag{3.13}$$

the degree of the functions is increased successively by adding control points until reaching the prescribed degree d.

Given one set of control points and the B-spline degree, the B-spline can be constructed. Nonetheless, in the proposed algorithm, the knots are the zero-level set samples of the SVM and the control points must be deduced. Given the set of knot points $\mathbf{P}$, the *interpolation of spline* determines a smooth B-spline approximation of degree d on the interval considered. The shape of the B-spline is conditioned by interpolation constraint equations. The B-spline end conditions used in the algorithm are discussed in subsection 3.5.4. To fit the input data points, the parametric curves are evaluated locally. A passing matrix $\mathbf{\Phi}$ allows linking the discontinuity knot points to the control points describing the B-spline [5] such as

$$\mathbf{Q}_{x/y} = 6\mathbf{\Phi}^{-1}\mathbf{P}_{x/y}, \tag{3.14}$$

where $\mathbf{P}_x$ and $\mathbf{P}_y$ are the x and y coordinates of the knots. The outputs of the algorithm are the control points $\mathbf{Q}_x, \mathbf{Q}_y$ of each B-spline. In the purpose of this report, cubic splines are considered.

Modifying one control point impacts only the partition of the segment that is concerned by this change. The number of patches affected by one control point depends on the degree of the polynomial. For cubic spline, four patches are affected when one control point changes [5]. The other way around, the polynomial description of one patch is influenced by four control points. Therefore, using B-spline allows local control.

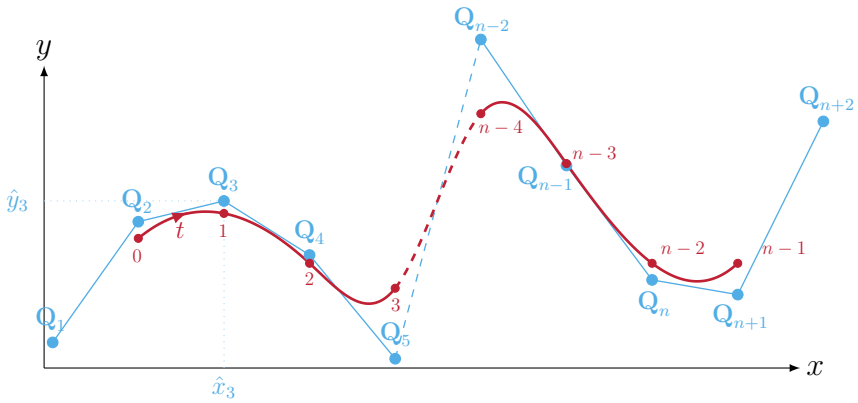


Figure 3.13: B-spline (——) defined by the control points $\mathbf{Q}_i$ (•) [5].

3.3.3 Local refinement (Block F)

The local refinement step consists in adding points at the most relevant place to improve the SVM classifier. The approach relies on the distance between the training labeled points of the SVM and the B-spline knot points. The uncertainty sampling method could have been used, so that the new points would have been randomly added where the approximation of the SVM is the poorest (the method used by Gorodetsky [33]). Nonetheless, the method proposed here avoids clusters of evaluated points in a single area, which is essential in the context of complex engineering models with long computation times.

The position of the new points relies on the distance between the knots ((•) in Figure 3.12) and the labeled points used to construct the SVM classifier ((•) annd (•) in Figure 3.12). As illustrated in Figure 3.14, the algorithm searches, for each knot point, the closest points of each class and computes the corresponding distance. The couple of points that have the

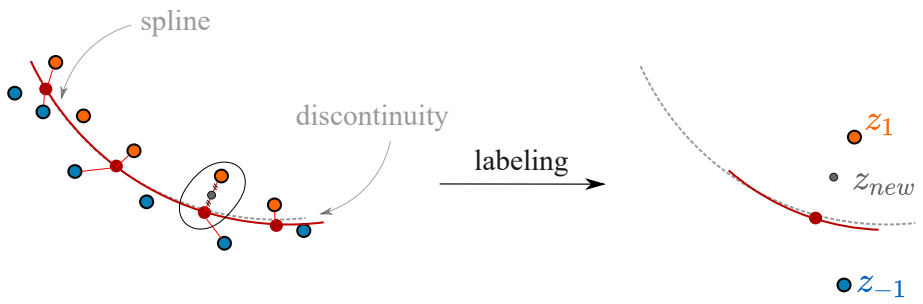


Figure 3.14: Schematic representation of the local refinement process.

highest length is retained, and a point is added in the middle of this edge. Each time a point is added, the response of the model is evaluated. A label is assigned to the latest evaluated point, depending on the proximity of its evaluation and the evaluation of the two closest training points of the SVM, such as

$$\begin{aligned} &\text{if} \quad |z_{new} - z_{-1}| < |z_{new} - z_1| \Rightarrow -1 \\ &\text{else} \quad \Rightarrow 1, \end{aligned} \tag{3.15}$$

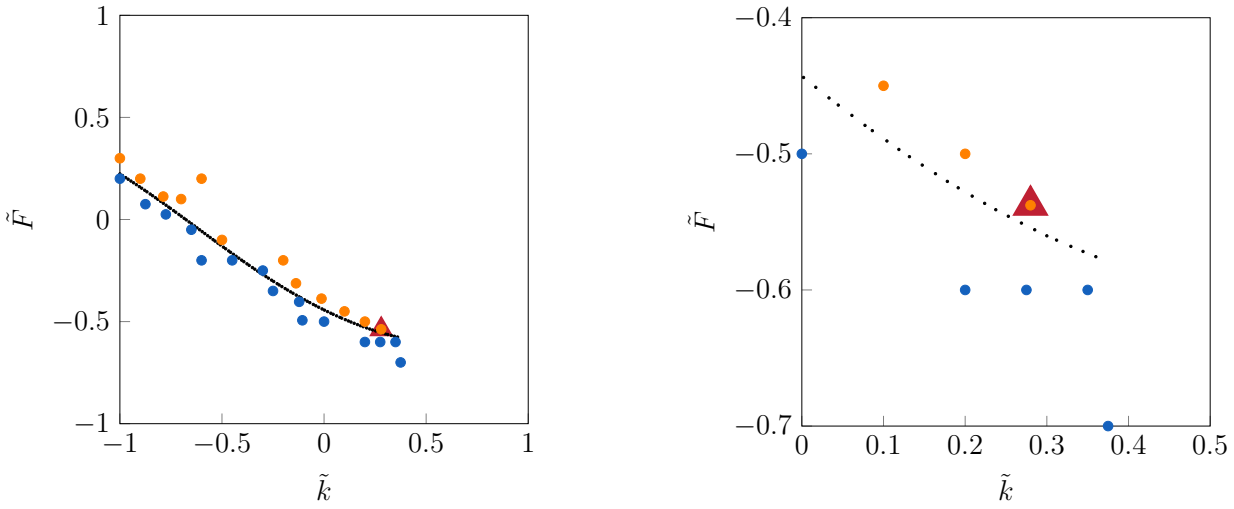
using the notations of Figure 3.14, right. A new SVM model is then established, and the process is repeated.

To limit computation times, several points could be added to the model during one iteration of the local refinement. This way, the evaluation of the model can be processed in parallel, reducing computation times. In addition, adding multiple points all at once reduces the number of SVM models to be constructed, also saving computational cost. In this report, the points are added one by one but adding a set of 10 points, for example,

could be interesting.

The process of local refinement can be stopped in two ways. If the total number of points prescribed by the user N_{max} has been reached, the process stops. Besides, when the localization of the discontinuity is considered sufficiently stable, convergence is reached, and the addition of new points stops. If multiple discontinuities are present, each of them is refined until the convergence criteria are reached. It is possible that one discontinuity still requires local refinement, while another one is stabilized. In that case, the algorithm stops adding points to the one that has converged and focuses on the less accurate one. A convergence criterion is required to study the convergence of the method. This is the outline of the next section. The total number of points added during the local refinement is named N_{local}.

The first iteration of the local refinement for the Duffing oscillator is shown in Figure 3.15. The new point (▲) is added in an interesting zone and the label (●) assigned is correct. A new SVM classifier has been trained (——).



(a) General view

(b) Zoom

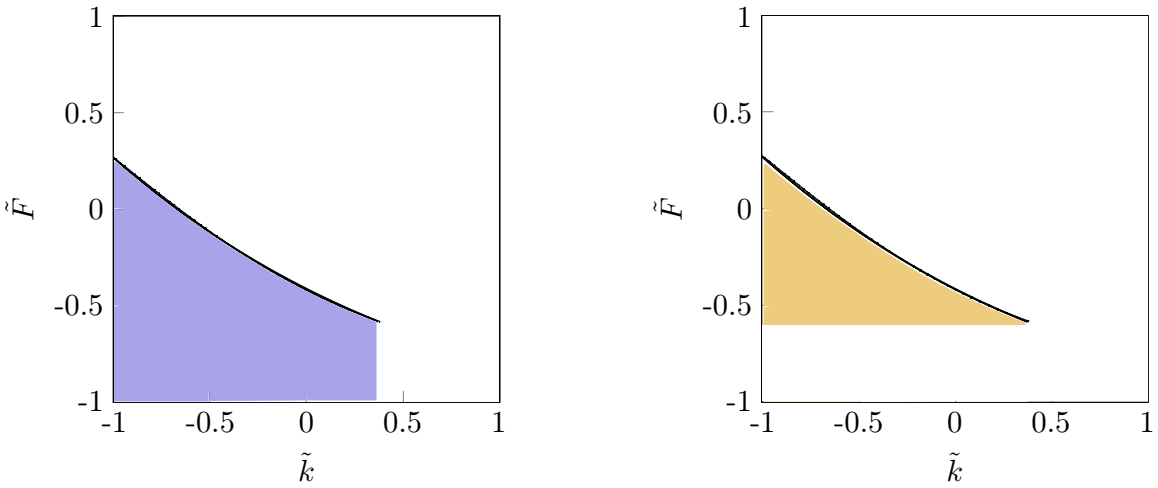
Figure 3.15: Nodes of the global refinement and the first added point of the local refinement (▲), with their corresponding labels: (●) class 1, (●) class -1.

3.3.4 Convergence criteria (Block G)

The greater the number of added points, the better the solution should be. Nonetheless, each point increases computation times. For that reason, a compromise between computational cost and accuracy must be made. The global refinement already provides a good overview of the discontinuity location, then the local refinement improves the accuracy of the localization, providing more training points to the Support Vector Classifier. The

B-splines obtained should not vary when lots of points are added. To assess this convergence of the localization of the discontinuities, stability criteria are introduced.

The criteria must be polyvalent. They should be independent of the orientation of the discontinuity and should be applicable to closed splines. For that purpose, two variables describing the B-spline are introduced: 1) the area under the B-spline along x A_x and 2) the area under the B-spline along y A_y, as illustrated in Figure 3.16.



(a) A_x (b) A_y

Figure 3.16: A_x ( ) and A_y ( ) for the Duffing oscillator.

The parametric curve coordinates can be written as a function of the intrinsic parameter t, such as

$$x = f(t) \qquad \text{and} \qquad y = g(t), \tag{3.16}$$

and the areas A_x and A_y the curve are defined as

$$A_x = \int_a^b f'(t)g(t)dt \qquad \text{and} \qquad A_y = \int_a^b f(t)g'(t)dt, \tag{3.17}$$

where a and b are the beginning and end points of the curve. Numerically, integration is approached using the *quad* function of the *scipy* [3] module of *Python*, which uses a procedure from the Fortran library *quadpack*. The method relies on an adaptive quadrature. The integral of the function $f(x)$ is approximated using static quadrature rules on adaptively refined subintervals of the region of integration [32].

In order to define a convergence criterion that is independent of the size of the domain, computed areas A_x and A_y are normalized with respect to the area of whole the domain A, such as

$$\tilde{A}_{x/y} = \frac{A_{x/y}}{A}. \tag{3.18}$$

This way, the criteria rely on the portion of the total domain area under the curve along x and y, respectively. The variation $\delta_{x/y}$ of $\tilde{A}_{x/y}$ between iterations i to $i+1$ are computed, such as

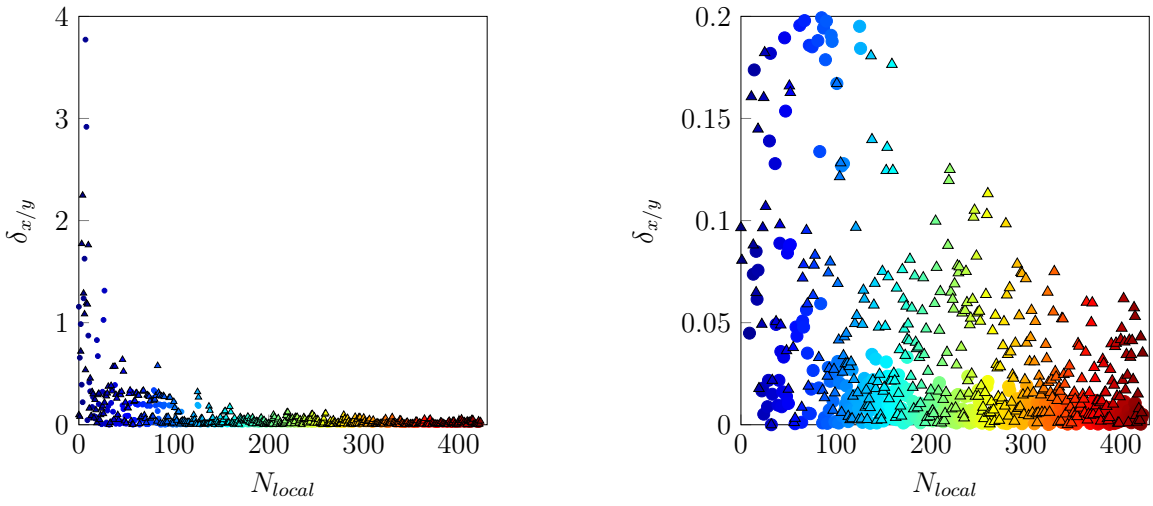
$$\delta_{x/y} = |\tilde{A}_{x/y}(i+1) - \tilde{A}_{x/y}(i)| \times 100. \tag{3.19}$$

These quantity characterizes the stabilization of the B-spline before and after adding a point during the local refinement. Indeed, the variation between two iterations will indicate the uncertainty on the B-spline obtained in the form of a percentage of the area of the whole domain. Thus the convergence threshold fixed by the user relies on the acceptable uncertainty on the discontinuity location in terms of an area. The local refinement stops when

$$\delta_x \ \ \& \ \ \delta_y < K, \tag{3.20}$$

with K the convergence threshold fixed by the user. The convergence is nonetheless limited due to different sources of numerical approximations, such as the precision of the integration method. Besides, adding lots of points nearby the discontinuity will not be useful for the approximation of the subdomain with polynomial chaos. Therefore, the constant K should not be too small.

The convergence of δ_x (●) and δ_y (▲) as a function of the number of points evaluated during the local refinement N_{local}, for the Duffing oscillator, are illustrated in Figure 3.17. The color gradient is related to the number of points added during the local refinement. The

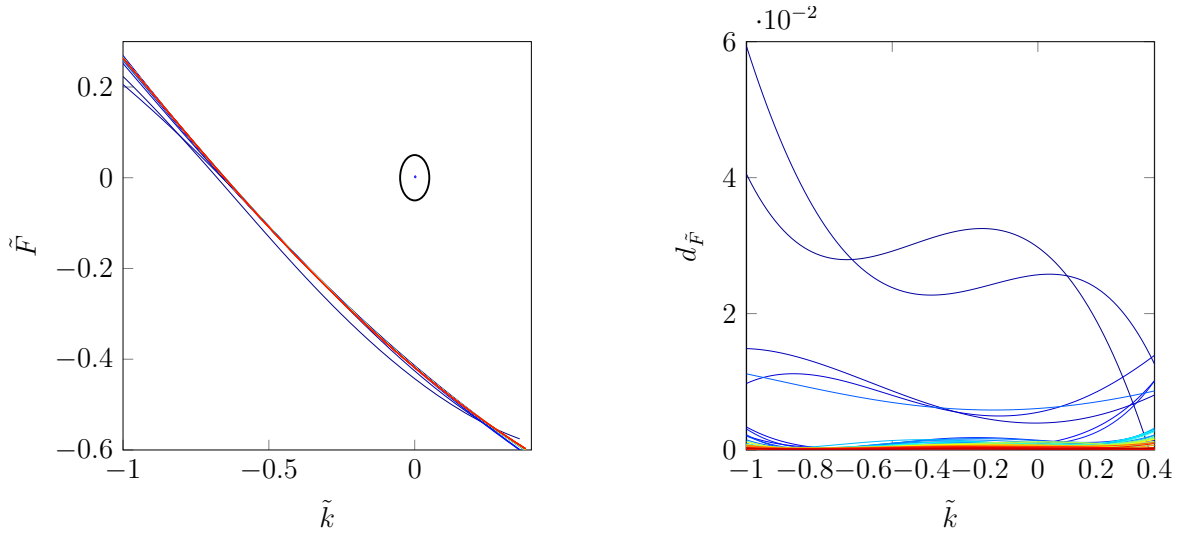


(a) General view (b) Zoom

Figure 3.17: Variation of the convergence criteria δ_x (●) and δ_y (▲) for the Duffing oscillator.

same color gradient is used to represent the B-splines obtained every 10 points added during the local refinement in Figure 3.18(a). At the beginning of the process, the curve varies a lot. Then, the curve stabilizes. The values of δ_x and δ_y rapidly decrease under 0.2%. This

means that the B-splines vary in an area equal to 0.2%A, since $A = 4$ for a domain bounded in [-1,1], the B-spline varies in a zone with an area equal to 0.008. Considering that the length of the B-spline is 1.67, this corresponds to a distance around the B-spline of 0.0048 since the variation of the area is distributed along the whole B-spline. These variations are not visible to the naked eye, as seen in Figure 3.18(b). The blue curves vary but then the other colors superimposed and only one red discontinuity is visible, meaning that the local refinement process allows stabilizing the discontinuity location. The differences $d_{\tilde{F}}$ in y (or in $\tilde{F}$ considering the Duffing oscillator) coordinates between the final curve and the curve at each iteration along x (respectively along $\tilde{k}$) are illustrated in Figure 3.18(b), using the same color gradient. The five first curves are far from the final one, but after 5 iterations of the local refinement, the difference $d_{\tilde{F}}$ decreases below $4 \cdot 10^{-2}$. This value corresponds to a portion of the domain and it is represented by the circled blue point in Figure 3.18(a).



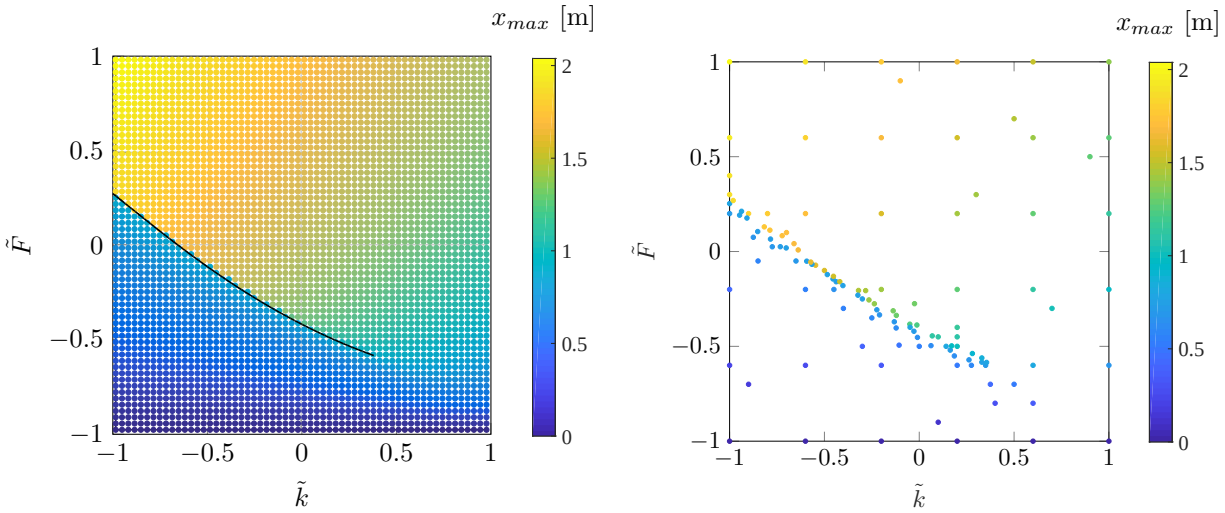
(a) Superimposition of the B-splines obtained each 10 iterations of the local refinement

(b) Difference of $\tilde{F}$ between the curve obtained each 10 iterations of the local refinement and the final curvel refinement.

Figure 3.18: Convergence of the discontinuity localization as a function of the local iterations.

3.3.5 Results

The localization of the discontinuity (——) of the Duffing oscillator obtained with the proposed algorithm is illustrated in Figure 3.19(a). The result is superimposed with the reference solution in order to support the discontinuity location. The outputs of the algorithm are the control point $\mathbf{Q}_x$ and $\mathbf{Q}_x$ of the B-spline (——). The numerical parameters used to reach this detection are given above in the different subsections. In particular, the global refinement termination threshold K was fixed to 0.1. The number of model evaluations used in the different steps of the algorithm are summarized in Table 3.4 and the corresponding evaluated points are illustrated in Figure 3.19(b).



(a) Final discontinuity location obtained with the proposed methodology

(b) N_{total} evaluated points

Figure 3.19: B-spline (——) obtained after $N_{total} = 123$ evaluated points, superimposed to the reference solution of the Duffing oscillator.

N_{basic}	N_{global}	N_{local}	N_{total}
46	29	48	123

Table 3.4: Number of model evaluations N at each step of the proposed methodology.

The main innovations of the proposed algorithm allow the localization of discontinuities that run partially through the domain, as illustrated with the Duffing oscillator. The proposed methodology requires a number of evaluations of the model ($N_{total} \sim 150$), which is similar to the number of points evaluated with the methods of Gorodetksy and V. Halder [36, 33] presented in subsection 2.3.5 (see Table 2.1). In addition, the accuracy of the localization is similar to the methods of these two authors, as it is justified later in section 4.1 when the method is applied to analytical models. Now that the overall methodology has been described, some numerical procedures introduced to improve the algorithm efficiency and robustness are described. Then, the sensitivity of the discontinuity localization algorithm to some of its numerical parameters is investigated.

3.4 Improvement of the algorithm efficiency

Different numerical procedures are used to improve the algorithm efficiency. First, the local re-meshing process is described. Then, the usefulness of multicore processing is highlighted.

3.4.1 Local re-meshing

An incremental Delaunay triangulation allows adding points one by one so that the grid is locally refined. Each time a point is added to the grid, a new tessellation is generated. Nonetheless, when a point is added to the set of points that correspond to a Delaunay mesh, only the triangles of which the circumscribing circle contains this point are suppressed [29] and new triangles are constructed locally, as illustrated in Figure 3.20. Using this property, only the gradients of the new triangles are computed, increasing the generality and the robustness of the code.

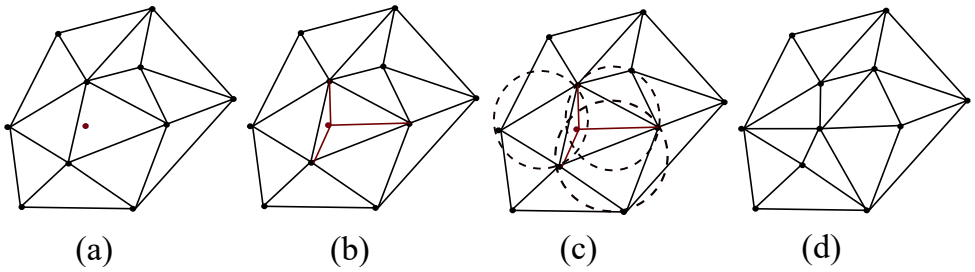


Figure 3.20: Local re-meshing of the Delaunay triangulation after addition of a new point (•) [42].

The main problem encountered when re-meshing the model with an additional point is the sequence of triangles that is completely changed. To keep some consistency between the gradients computed for each triangle and the triangle numbering, a dedicated numerical procedure is implemented. Each point of the mesh is numbered (from 0 to the number of points N_{total}). Each time a new point is added, it is added to the end of this list. By definition, a triangle is completely described by its three vertices, and each vertex corresponds to one point of the mesh. Therefore, each triangle is completely characterized by three numbers a_i, b_i and c_i, corresponding to the vertices numbers. This way, a code χ_i is assigned to each triangle i, such as

$$\chi_i = a_i 10^6 + b_i 10^3 + c_i \qquad i = 1, \ldots, N_\triangle, \tag{3.21}$$

where a_i is the largest value of the three vertices, c_i the minimum, and b_i the middle one and $N_\triangle$ the number of triangles built during the procedure. For each code χ_i corresponding to a given triangle, the norm of the gradient is computed. To this extent, when a new

point is added to the mesh, a code is assigned to the newly created triangles, and only the corresponding norms of the gradient are computed.

The values 10^6 and 10^3 in Equation (3.21) must be carefully chosen since they limit the number of points that can be added to the domain. Considering these values, the maximum number of points N_{max} is 999. Indeed, larger values for a, b and c would lead to overlapping so that the triangles would not be uniquely identified. Depending on the maximum number of points N_{max} specified by the user, the coefficients 10^6 and 10^3 are thus adapted.

The same reasoning is applied for the computation of the difference between the evaluations of the model at two points. These values are indeed used in the global refinement step and to compute the gradient of the edges. Since all edges are not modified when points are added during the global refinement, the values should not be all recomputed. This time, one edge is completely defined with two vertices a_i and b_i. Therefore, a code Ψ_i is assigned to each edge i, such as

$$\Psi_i = a_i 10^3 + b_i \qquad i = 1, \ldots, N_{edges}, \tag{3.22}$$

where a_i is the largest value between the two vertices number, and b_i is the lowest one, and N_{edges} is the total number of edges constructed during the process. Each time a new code is created, in other words, when a new edge appears in the mesh, the difference between the evaluations of the function at these two vertices is computed.

The use of these specific numerical procedures hasn't introduced any change in the global mesh obtained, given the same initial conditions. This proves that the results are not impacted by the local re-meshing procedure, but it introduces improvement from an algorithmic point of view. Indeed, it introduces consistency in the triangles numbering during the whole procedure and possible feedback on the previous iteration. Regarding computation times, the methods have led to 1.5% of time saving, which is not very significant over computation times of $1.5 \cdot 10^{-2} s$ per iterations. Nonetheless, this can further reduce computation times for meshes composed of a large number of cells.

3.4.2 Multicore processing

The second numerical tool used to improve the algorithm efficiency is the multicore processing, which has an important impact on computation times. Multiprocessing consists in processing two or more different portions of the set of instructions simultaneously [6].

The identification of the position of the new points must be made in sequential given the process used. Nonetheless, the evaluation of the added points can be executed in parallel,

reducing computation times. The time saved is directly linked to the time required to evaluate the response of one point. In the case of computationally expensive engineering models, such as the industrial application of blade/casing interaction presented later in the report, using parallel processing is particularly useful since several points can be evaluated all at once. Nonetheless, it is not recommended to add more than two points all at once during the global refinement. Indeed, the new mesh and the new triangle characteristics are computed after all the points have been evaluated simultaneously. Therefore, the risk is to get clusters of points in a zone and that could affect the global refinement process. In addition, the algorithm can take advantage of parallel processing to construct the basic mesh. Besides, computing the triangles gradients can also be executed in parallel. Finally, during local refinement, if multiple points are added, they can be evaluated all at once as well, improving the numerical efficiency.

3.5 Sensitivity to numerical parameters

In the last section of this chapter, the sensitivity of the proposed discontinuity localization tool to some of its numerical parameters is studied. The impacts on the discontinuity location of four main numerical parameters are investigated: the basic mesh size, the global refinement mesh size, the parameters of the SVM, and the end conditions of the B-spline.

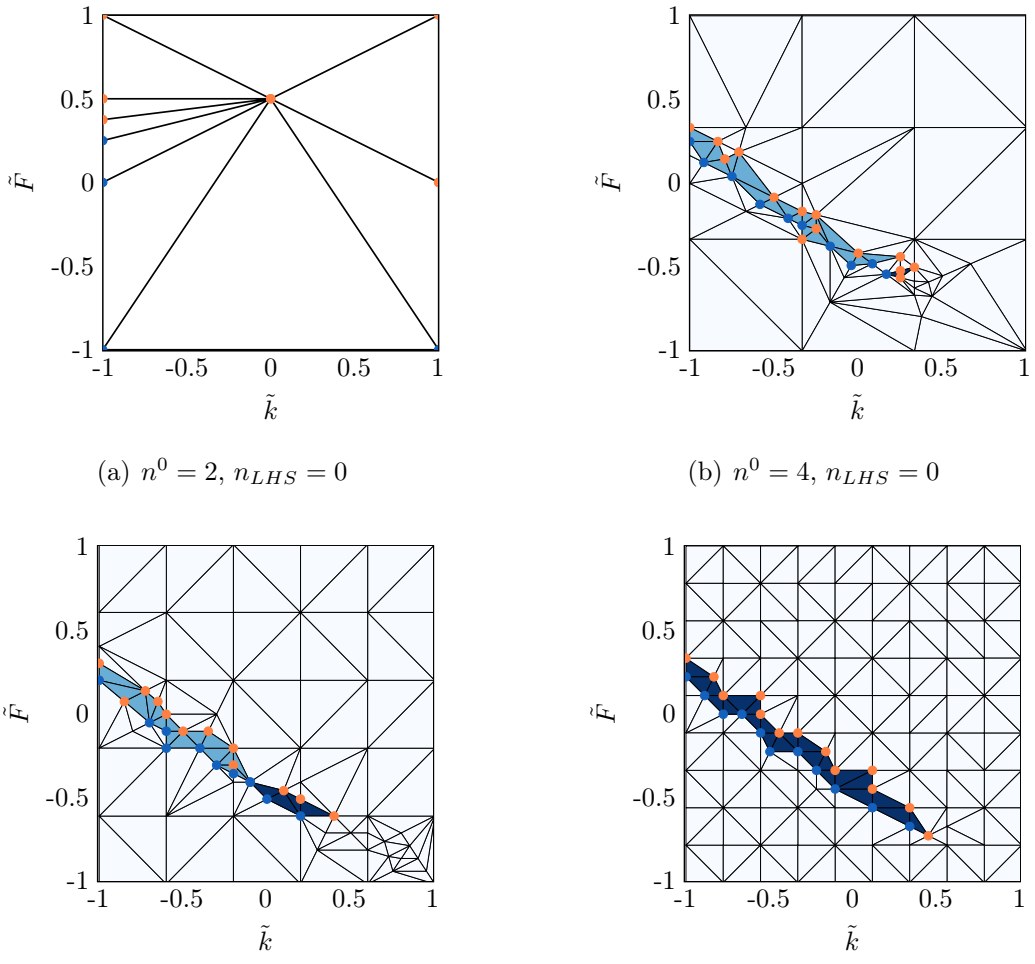
3.5.1 Basic mesh nodes (N_{basic})

The number of nodes N_{basic} of the basic mesh has a direct impact on the global refinement process. To evaluate the dependence of the process to this parameter, different mesh sizes are tested, and the global meshes obtained at the end of the global refinement are illustrated in Figure 3.21. The usefulness of the n_{LHS} Latin hypercube nodes is also discussed based on Figure 3.22. The lowest possible value for the number of points of the basic regular grid n^0 is 2, to have at least one element in the mesh as illustrated in Figure 3.21(a).

Firstly, a structured basic mesh is considered, so that no Latin hypercube nodes are added to the basic mesh, and different sizes of the regular meshes are considered: $n^0 = 2, 4, 6$ and 10. It may be noted that the algorithm allows using different numbers of nodes n^0 on the x and y axis if the user so wishes. The global mesh and the detected discontinuities are illustrated in Figure 3.21, where the labeled points (●) and (●) are represented. For $n^0 = 4$ and $n^0 = 6$, two discontinuities are detected (the first one ▲ and the second one ▲) as seen in Figure 3.21(b) and (c), but only one was expected. When only four nodes are used ($n^0 = 2$), the basic mesh has only one cell as shown in Figure 3.21(a) and no discontinuity area has been detected. The global refinement process stops when only 6 points are added because the triangle characteristics are not distinguishable. Finally, using a larger number of basic

points such as $n^0 = 10$ (Figure 3.21(d)) has led to a correct global refinement, but this is not in accordance with the minimization of the evaluated points. Therefore, a good compromise would be to use a regular grid with $n^0 = 6$.

Then, as seen in Figure 3.22, $n_{LHS} = 10$ Latin hyper-cube nodes are added to the regular mesh, leading to an unstructured basic mesh. The same values of n^0 are considered: $n^0 = 2, 4, 6$ and 10. The improvement of the results, thanks to the addition of the randomly distributed points, is illustrated in Figure 3.22. Indeed, only one discontinuity is detected instead of two in the previous case, and the case $n^0 = 2$ with Latin hypercube nodes shows the desired result. This justifies the use of the Latin hyper-cube nodes rather than using a regular grid alone. Indeed considering a regular mesh without Latin hypercube nodes has shown a good global mesh only for $10n^0 \times 10n^0 = 100$ basic points. In contrast, the addition of $n_{LHS} = 10$ Latin hypercube nodes shows satisfactory global mesh using only $2n^0 \times 2n^0 + 10n_{LHS} = 14$



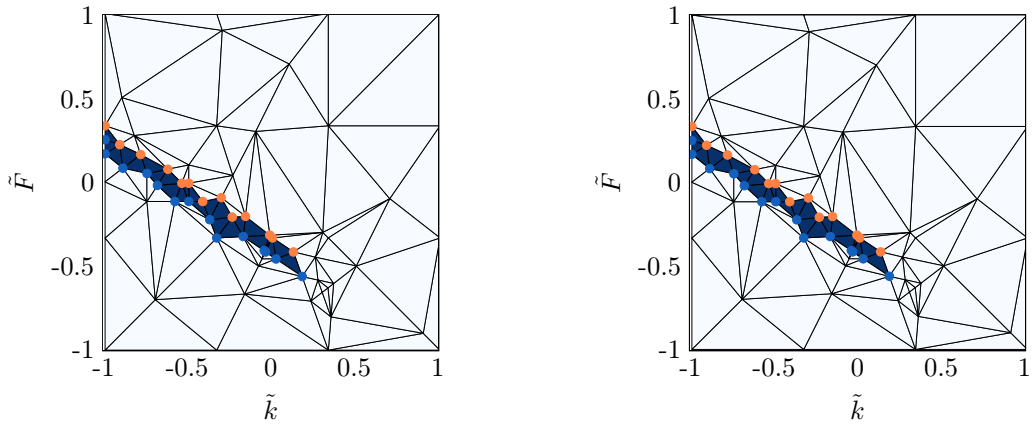
(a) $n^0 = 2$, $n_{LHS} = 0$

(b) $n^0 = 4$, $n_{LHS} = 0$

(c) $n^0 = 6$, $n_{LHS} = 0$

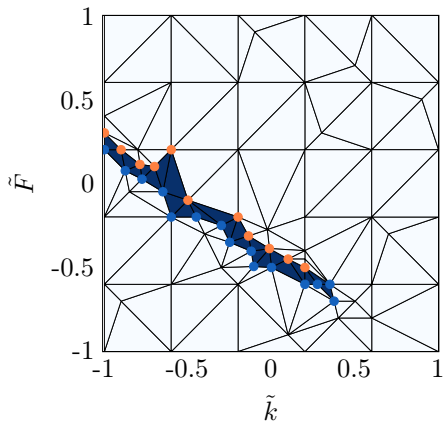
(d) $n^0 = 10$, $n_{LHS} = 0$

Figure 3.21: Global mesh and discontinuity triangles for different basic mesh nodes n^0, without Latin hypercube node $n_{LHS} = 0$.

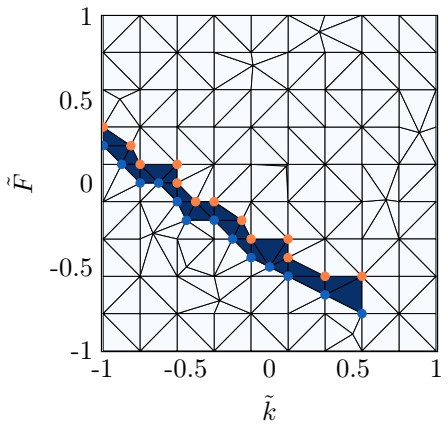


(a) $n^0 = 2$, $n_{LHS} = 10$

(b) $n^0 = 4$, $n_{LHS} = 10$



(c) $n^0 = 6$, $n_{LHS} = 10$



(d) $n^0 = 10$, $n_{LHS} = 10$

Figure 3.22: Global refinement and discontinuity triangles for different basic mesh nodes n^0, with $n_{LHS} = 10$ Latin hypercube nodes.

basic points. Nonetheless, using randomly distributed nodes introduces uncertainty in the algorithm behavior, which makes it difficult to assess the convergence of the procedure, requiring stochastic developments.

3.5.2 Global refinement: mesh size (ε)

During the global refinement, the minimal distance ε between two evaluated points of the domain is controlled by the user with the variable e which fixes ε (see subsection 3.2.2, Equation (3.7)). The impact of the minimal distance between two nodes on the convergence of the results is explored in this section.

First of all, the minimal distance is upper and lower bounded. The minimal distance cannot be larger than the basic mesh size, since no refinement would be possible in that case. Considering a basic grid of $6n^0 \times 6n^0 = 36$ points, the minimal distance between two

points of the domain must be at least below half of the largest edge size. In addition, if the distance is too large, there is a risk to badly detect the discontinuity triangles. For example, if two discontinuities are close to each other, the method will detect one zone of discontinuity instead of two distinct ones. However, a too fine global refinement, meaning a very small value of ε, could cause the detection of discontinuity triangles to wrongly operate and add many points nearby the discontinuity, which is useless for the approximation method following the detection.

In this context, the impact of ε is studied, with e varying in the range $e \in [2; 14]\%$, meaning that the minimal distance between two nodes ε thus varies in the range $\varepsilon \in [0.04; 0.28]$ considering the Duffing oscillator case. For visualization purpose, the largest and lowest distances are illustrated in Figure 3.23. Indeed, for the reasons mentioned just above, larger or lower values are not recommended for successful operations of the algorithm.

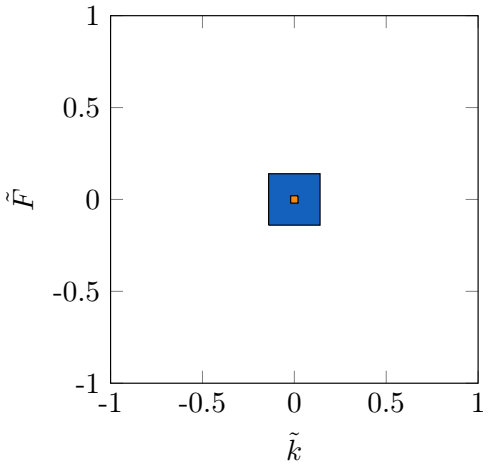


Figure 3.23: Squares of edges 0.04 (■) and 0.28 (■), the largest and lowest studied values of ε.

Two aspects must be analyzed. First of all, the global mesh process is uncertain due to the randomly distributed nodes, as stated in the previous section. Therefore, running the same simulation with different Latin hypercube nodes leads to different global mesh. The value of the minimal distance between two points has an impact on the proper functioning of the global refinement step. For large distances between two points, namely $\varepsilon > 0.12$, the labeling process will not behave as expected because few points will be added during the global refinement. Therefore, the labels assigned to the points already present in the basic mesh have a larger risk of not being properly labeled, which could badly influence the construction of the Support Classifier. Therefore, it is not recommended to fix $\varepsilon > 0.12$ in order to get a good global refinement. Indeed, the process could behave correctly considering some basic mesh, but the stability of the discontinuity detection is very limited.

The second impact of the minimal distance between two nodes concerns the local refinement process. As shown in Table 3.5, the number of points evaluated during the global refinement does not vary much for the four largest values of e. Therefore, the number of

ε	0.28	0.24	0.2	0.1	0.12	0.08	0.04
N_{global}	9	9	8	9	11	29	49

Table 3.5: Number of points N_{global} evaluated during the global refinement for different values of ε.

nodes evaluated during the local refinement is studied to determine which size gives the best performance. For that purpose, the algorithm is run for different values in the given range of ε, and results that have given good global meshes are analyzed. To study the convergence, the stabilization criteria $\delta_{x/y}$ is studied in the range of 90 to 100 points added during the local refinement. The maximum value of the convergence criteria $\delta_{x/y}$ in this range is retained for each value of e and the results are illustrated in Figure 3.24. As expected, adding points increases the accuracy of the detection, but the localization of the discontinuity should require as few points as possible to be efficient. Therefore, this graph allows making a good compromise between computation times and accuracy, comparing the total number of evaluated nodes and the convergence of the stabilization criteria. Actually, the main trend illustrated in Figure 3.24 is the linear behavior between the number of points N_{total} evaluated during the process and the accuracy of the discontinuity localization.

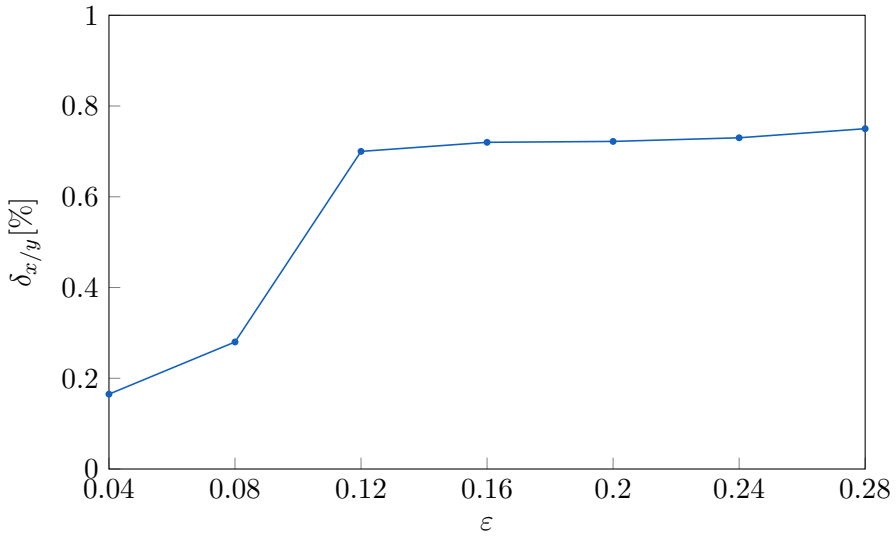


Figure 3.24: Maximal value of $\delta_{x/y}$ between 90 and 100 iterations of the global refinement as a function of the minimal distance between two points ε.

3.5.3 SVM classifier parameters ($C, \gamma, kernel$)

Support Vector classifiers are widely applied in the field of pattern classification and their main advantage is the fact that the method is suitable for both linearly and nonlinearly separable data. Nonetheless, one drawback of SVM is that the input parameters must be adjusted depending on which type of geometry occurs. For that purpose, the module *model_selection* of Python is used to choose which parameters are the best suited for the current discontinuity location shape.

The method constructs a grid combining each possible combination of parameters summarized in Table 3.3 and trains a SVM classifier with each set of parameters. A score is then computed for each SVM classifier and the set of parameters given the best score is retained to construct the final SVM classifier.

An example of the impact of the used Kernel function and C parameter on the score of the classifier is illustrated in Figure 3.25 and 3.26, considering the Duffing oscillator. First, regarding Figure 3.25, two possibilities of Kernel functions have been used: 1) a polynomial of degree 3 (■) and 2) a Gaussian rbf Kernel with $\gamma = 10$ (●). Searching for a model more complex than the discontinuity itself can lead to a bad representation of the discontinuity [14]. This claim is well illustrated in Figure 3.25. Indeed, the score of the SVM classifier is lower when a Gaussian rbf Kernel is used because the geometry of the discontinuity to be detected is not so complicated. In addition, adding points during the local refinement does not improve the score. On the contrary, the score decreases, meaning that the Gaussian rbf Kernel is badly adapted. Another limitation of the SVM is that it is not well suited to approximate sharp edges boundaries [36], whatever the Kernel function used. Nonetheless, in the context of nonlinear behavior occurring in blade/casing contact interaction, sharp corner geometries are normally not the most encountered shapes.

The impact of the parameter C is illustrated in Figure 3.26, where $C = 10$ (○) and $C = 100$ (■) are considered. By definition of the regularization parameter C, the score of the SVM is lower when $C = 10$. Nonetheless, for more complicated shapes, the risk of using large C parameters is to get overfitting and thus an oscillating separator hyperplane rather than a smooth curve.

3.5.4 B-spline end conditions ($\mathbf{\Phi}$)

For the same distribution of knots, different control points can be obtained from Equation (3.14), depending on the type of end conditions fixed. The coefficients of the matrix $\mathbf{\Phi}$ are different depending on the chosen type, so that the shape of the curve at both ends is adapted. Four main possibilities exist for the end curve conditions. In the context of this work, *free*

end conditions and *end-to-end tangent continuity* are considered [5].

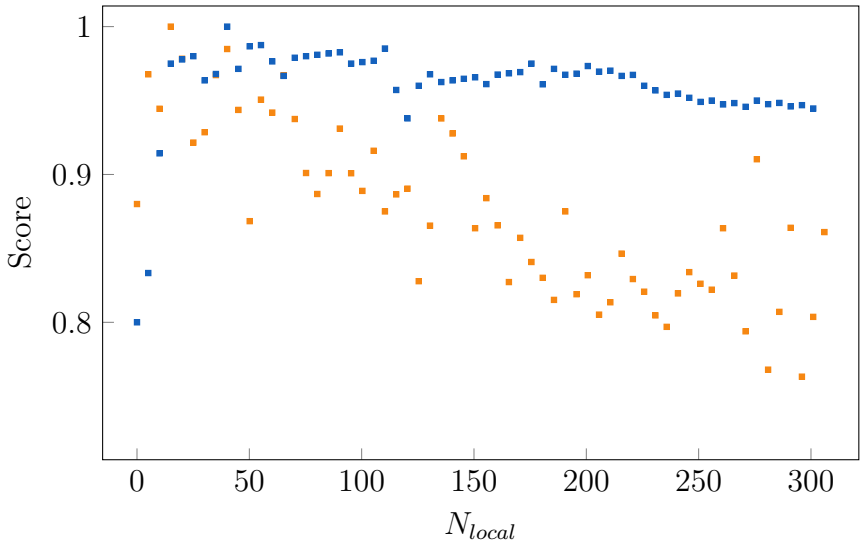


Figure 3.25: Evolution of the score of the SVM with the number of points added during the local refinement N_{local} using two different Kernel function: 1) a polynomial of degree 3 (■) and 2) a Gaussian rbf kernel with $\gamma = 10$ (●) and $C = 100$.

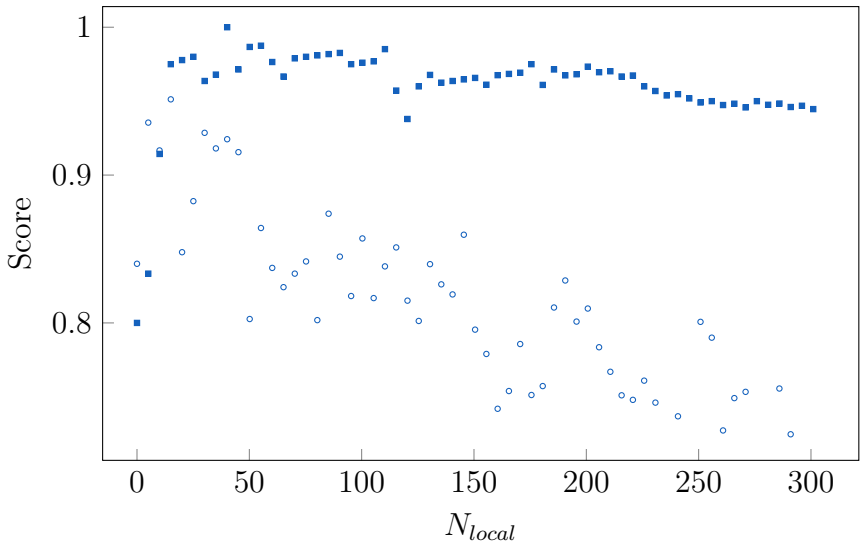
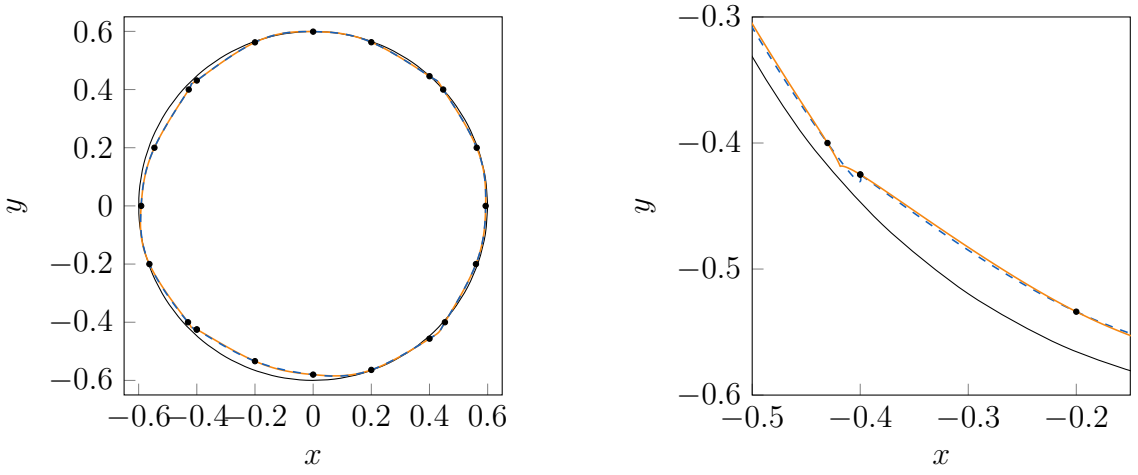


Figure 3.26: Evolution of the score of the SVM with the number of points added during the local refinement N_{local} using a polynomial of degree 3 as Kernel function and to values for the parameter C: 1) $C = 100$ (■) and 2) $C = 10$ (○).

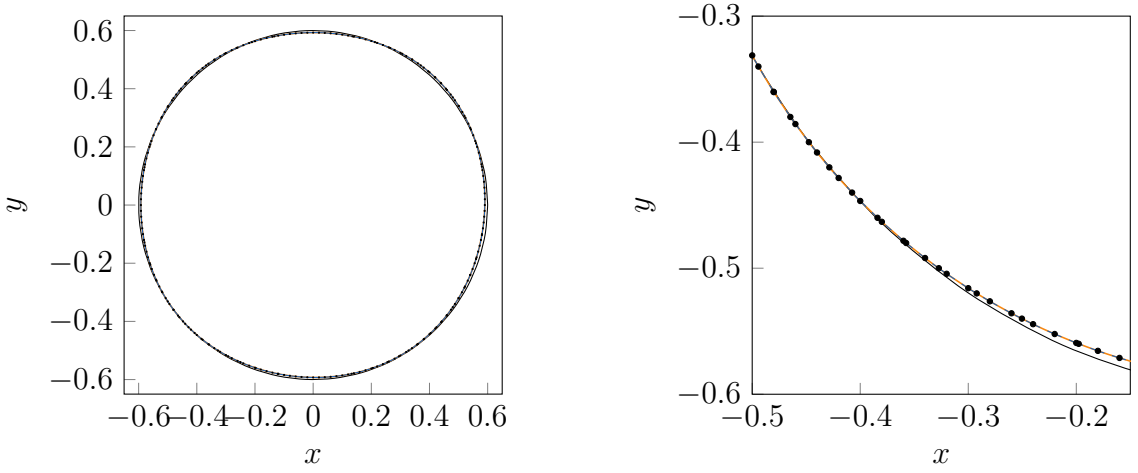
The free end condition imposes the three first and last control points to be aligned and separated two by two by the same distance. This allows fixing the direction of the tangent, which corresponds to the line formed by the three endpoints at each side of the spline. Besides, end-to-end tangent continuity imposes the same slope at first and last fit knots. Linking the parametric curves at both ends is useful when closed geometries are considered.

One of the examples presented in section 4.1, namely the circular discontinuity, is used to illustrate the impact of the B-spline end conditions in the proposed method. The B-splines obtained with free end conditions (- - - -) and end-to-end tangent continuity conditions (——) are studied considering different densities of knot points. The B-splines are illustrated in Figure 3.27 and 3.28 respectively, where the knots (•) obtained from the SVM classifier are represented. The exact position of the discontinuity (——) is also represented. The number and the spacing of knots are directly linked to the size of the grid where the Support Vector classifier is evaluated. The case where knots are spaced (11 points on the SVM grid) is



(a) General view (b) Zoom

Figure 3.27: B-splines obtained with free end (- - - -) and end-to-end tangent continuity conditions (——), with a small density of knots (•) and the actual discontinuity location (——).



(a) General view (b) Zoom

Figure 3.28: B-splines obtained with free end (- - - -) and end-to-end tangent continuity conditions (——), with a large density of knots (•) and the actual discontinuity location (——).

illustrated in Figure 3.27, while the knots are very close to each other (101 points on the SVM grid) in the case of Figure 3.28. The end conditions have an impact on the B-spline obtained if the number of knots is small. The end-to-end tangent continuity is better suited for a closed shape, as illustrated in Figure 3.27(b). Indeed, an artifact appears at the junction between the end and the beginning of the spline when free end conditions (- - - -) are used. This cusp is due to the alternation between points far and near to each other, which particularly constrains the B-spline. On the contrary, the B-spline behaves better with end-to-end tangent continuity (——). Nonetheless, for a fine Support Vector Machine grid, the impact of the end condition is not visible, as illustrated in Figure 3.28. A fine grid is thus recommended. This way, free end conditions are considered, whatever the shape of the discontinuity. Computation times are not significantly impacted when a fine SVM grid is used, and this avoids the requirement to make a distinction between closed and open discontinuities. In addition, the distance between the discontinuity localized with the proposed algorithm and the actual circle (——) to be detected is much smaller when a dense SVM grid is used. Therefore, it is even more interesting to evaluate the SVM classifier on a dense grid.

3.6 Summary

The various numerical tools used in the proposed methodology have allowed the construction of a 2D discontinuity localization tool that can deal with closed discontinuities, multiple discontinuities, and jump running partially through the domain. To the best of the author's knowledge, no approaches found in the literature allow managing all these features together. In addition, the convergence of the discontinuity location is very precise considering a limited number of evaluated points, as expected. The main disadvantage of the proposed methodology is the dependence of the results on its numerical parameters, which must be carefully chosen to ensure good performances.

Now that the algorithm has been described, the proposed methodology is tested on various analytical models to illustrate its applicability. The methodology is then applied to localize a discontinuity in a numerically costly model in the context of blade/casing contact interactions. These are the contents of the last chapter.

4 | Numerical examples

In this last chapter, the applicability of the proposed discontinuity localization algorithm is demonstrated on a variety of analytical problems. Then, the algorithm is applied to the complex engineering model of interest in the context of blade/casing interactions.

4.1 Analytical models

Four analytical models, illustrated in Figure 4.1, are introduced to evaluate the pertinence of the algorithm on distinct geometries of discontinuities in the Design of Experiments. In particular, the applicability to localize multiple discontinuities is studied, as well as the possibility to distinguish jump and areas with a large slope. Through the different models, the pros and cons of the algorithm are highlighted. The main advantage of the analytical models (a), (b), (c) and (d) presented in this section is the ease of computing a reference solution in order to validate the localization of the discontinuity obtained with the proposed methodology.

The reference solutions $u(x, y)$ for the four models studied are illustrated in Figure 4.1. The models have been evaluated on a dense grid. The inputs are normalized and vary in [-1;1]. The discontinuities location in the xy plane are known analytically and read:

$$(a)\ \ 0.6^2 = x^2 + y^2; \tag{4.1}$$

$$(b)\ \ x = 0; \tag{4.2}$$

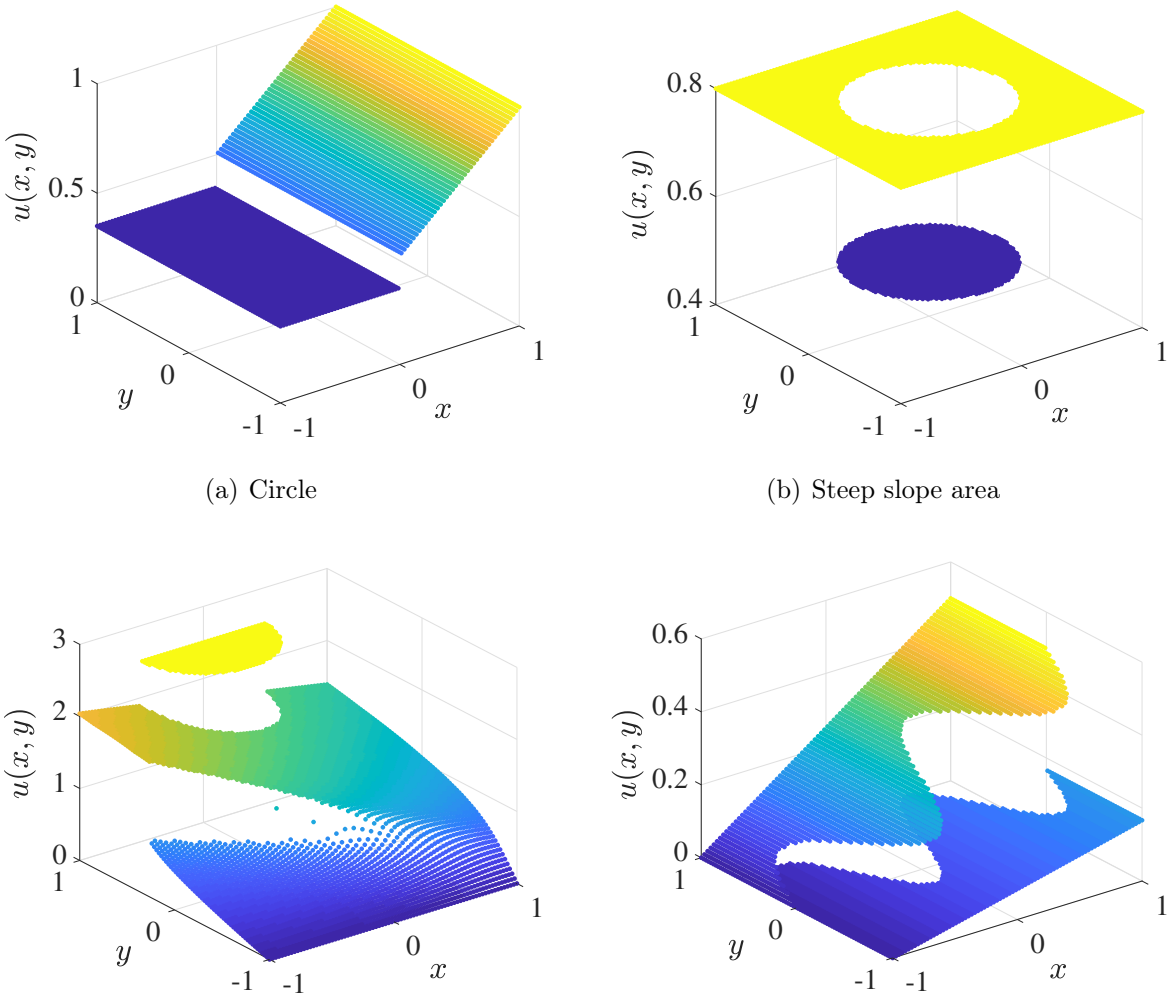
$$(c)\ \ 0.5^2 = x^2 + (y-1)^2; \tag{4.3}$$

$$(d)\ \ y = \frac{1}{8}\sin\left(2\pi(x+1)\right). \tag{4.4}$$

Various cases are tested by means of these four models. A discontinuity with a closed shape is considered with model (a). Besides, model (b) presents a zone of steep slopes. Multiple discontinuities localization is tested with function (c), and finally, a more complex discontinuity geometry is presented with the model (d). Case (c) has two discontinuities. Indeed, this model corresponds to the Duffing oscillator, studied in chapter 3, where a second discontinuity has been manually added. This second discontinuity is the one written in Equation (4.3). The exact discontinuity location of the duffing oscillator is not known. Therefore, it is not possible to compute the error with the exact discontinuity location for this jump.

The locations of the discontinuities in the xy plane obtained with the proposed detection algorithm are presented in this section. A basic mesh with $n^0 = 6$ points and $n_{LHS} = 10$ points is used for each case. Besides, the minimal distance between two evaluated points ε is fixed to 0.1 for each simulation. The parameters of the SVM used for the presented results are summarized in Table 4.1, and the local refinement termination threshold is fixed to $\delta_{x/y} < K = 0.1$. The numbers of evaluated points for each model, at each step of the algorithm, are summarized in Table 4.2, where N_{basic} is the number of nodes of the basic mesh, N_{global} the number of points added during the global refinement and N_{local} the points evaluated during the local refinement. The maximum number of evaluations is fixed to $N_{max} = 200$. For the first three cases, the process stopped adding points before reaching this limit.

The errors between the actual location of the discontinuity and the detected one,



(a) Circle

(b) Steep slope area

(c) Multiple discontinuities

(d) Sinusoidal shape

Figure 4.1: Reference solutions of the four proposed analytical models.

corresponding to the distance between both curves, are computed using the analytical functions of the curve and the parametric equation of the B-spline of the corresponding discontinuity location. The error is defined as the distance d between both curves. It is computed using cylindrical or Cartesian coordinates depending on the case considered.

Analytical functions	Kernel function	C	γ
(a)	Gaussian rbf	100	1
(b)	Linear	100	–
(c) Half-circle/Duffing	Gaussian rbf/Poly3	100/100	10/-
(d)	Gaussian rbf	10	10

Table 4.1: SVM parameters corresponding to the given solutions.

Analytical functions	N_{basic}	N_{global}	N_{local}	N_{total}
(a)	46	48	77	171
(b)	46	36	5	87
(c)	46	64	47	157
(d)	56	70	74	200

Table 4.2: Number of evaluated points for each model at the different steps of the algorithm.

(a) Circle

The discontinuity location obtained with the proposed algorithm (——) is illustrated in Figure 4.2(a). The curve has been superimposed to a grey and white background which illustrates the actual boundary between the two smooth subdomains. The localization error is computed using cylindrical coordinates θ. The distance d between the actual discontinuity location and the curve found with the proposed algorithm as a function of θ is illustrated in Figure 4.2(b). The mean distance $\bar{d}$ between the actual and the localized discontinuities can be computed and is equal to

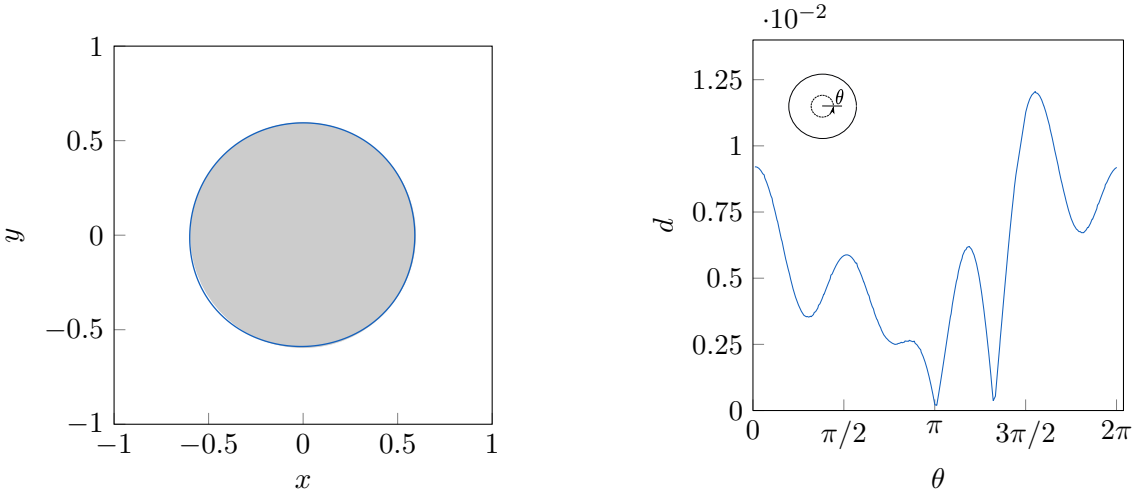
$$\bar{d} = 5.5 \cdot 10^{-3}. \tag{4.5}$$

Considering an error $\bar{d}$ distributed along the whole circle, the misclassified portion of the domain δA can be computed such as

$$\delta A = \frac{2\pi\bar{d}}{(x_{max} - x_{min})(y_{max} - y_{min})} = \frac{0.3456}{4} = 8.6 \cdot 10^{-3}. \tag{4.6}$$

The order of magnitude of the error obtained is the same as the accuracy of the results obtained by V. Halder *et al.* [36], whom proposed another method to localize discontinuities.

The required number of evaluation of the model is also similar to the number required with the methodology of these authors. The 171 points evaluated to reach this solution are illustrated in Figure 4.3.



(a) Localization of the discontinuity (——) obtained with the proposed methodology

(b) Distance d between the actual and the detected curves as a function of θ

Figure 4.2: Localization of the discontinuity with the proposed methodology and difference d with the actual discontinuity location for case (a).

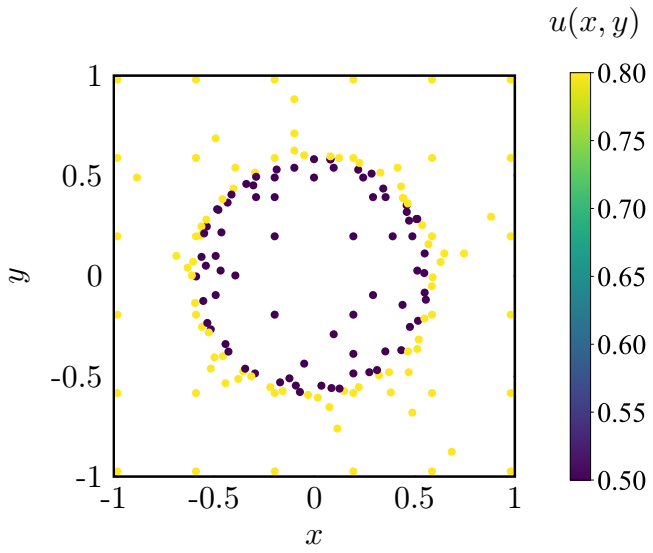


Figure 4.3: N_{total} points evaluated during the localization process for case (a).

The Latin hypercube nodes' positions vary from one run to another, modifying the global refinement results. To study the convergence of the local refinement independently of the location of the basic nodes, the algorithm is run ten times. For each run, the mean distance $\bar{d}$ is computed each time 20 points are added during the local refinement. The number of nodes

added during the global refinement varies in the range of [34;43] for the 10 simulations tested. To take into account the random distribution of the basic Latin hypercube nodes, the mean of $\bar{d}$ (——) is computed over the ten runs as a function of the number of points evaluated during the local refinement. The results are illustrated in Figure 4.4. The area within plus one and minus one standard deviation (——) of the mean is illustrated in Figure 4.4. The example of Figure 4.2 is identified by the black point (•). Overall, this illustrates that the basic mesh has an impact on the accuracy of the discontinuity location considering the results of the global refinement process. Then, the local refinement increases the correctness of the prediction of the discontinuity localization as expected. Nonetheless, the improvement tends to stabilize when more points are added during the local refinement. This limitation is mainly due to the fact that the circle is approximated using B-spline. In addition, the SVM classifier has a limited convergence as detailed in the next analytical case.

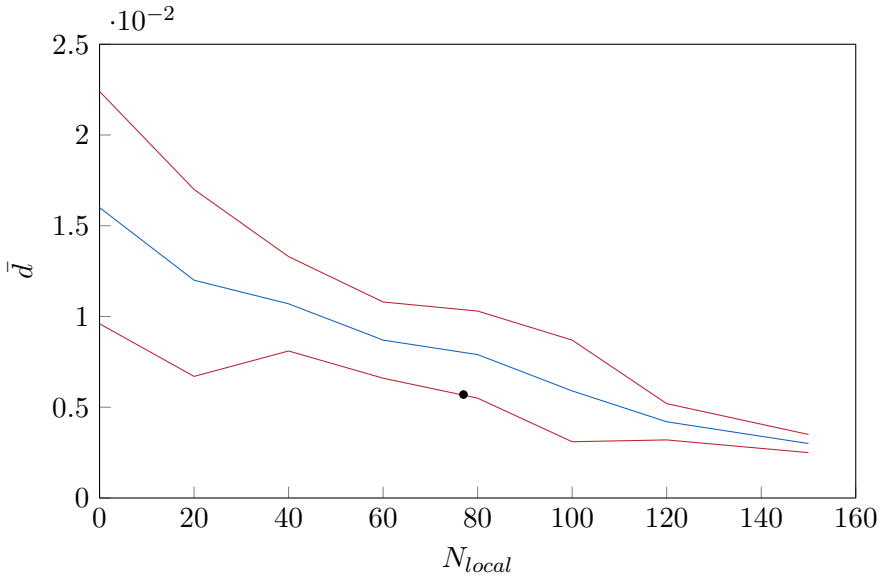
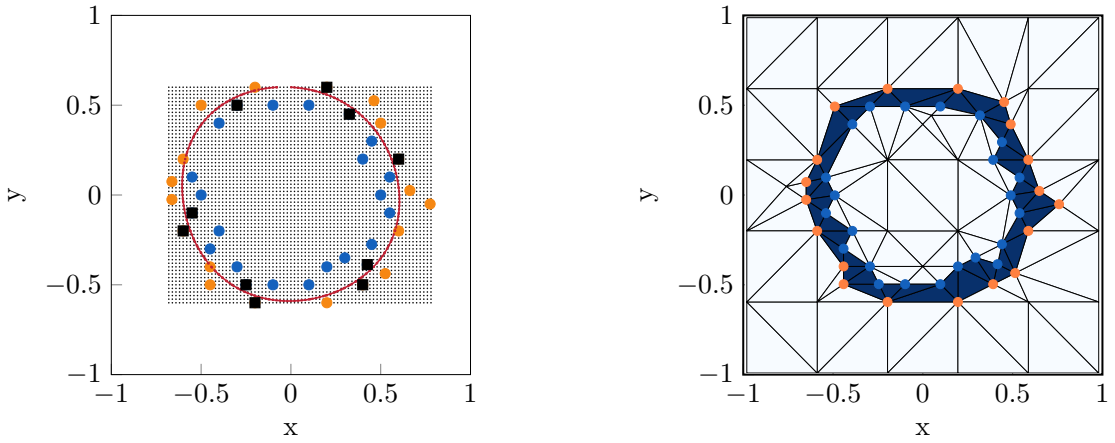


Figure 4.4: Mean variation distance $\bar{d}$ as a function of the number of points added during the local refinement N_{local}.

Closed discontinuities do not require specific treatment since the SVM classifier will automatically detect that the discontinuity is closed using the training points information. This way, the first and the last knots are the same, and the B-spline will be closed as well. Indeed, if the first and the last knots would be different, the process would have to detect that the spline is closed and should duplicate the first node. Nonetheless, as illustrated in Figure 4.5(a), the risk of using an evaluation grid bounded by the largest coordinate labeled points is that the zero level set of the classifier is outside the defined grid. In such a case, the discontinuity detected could be open on one side. In this example, a regular mesh is considered, meaning that no Latin hypercube nodes are added to the basic mesh. As seen in Figure 4.5(b), they are two labeled points in a same line in the upper part of the circle, and no labeled point in between, so that the SVM grid is limited by these points, causing

the SVM classifier to be potentially non-closed.



(a) Non-closed SVM classifier (——), and labeled samples: class -1 (●) and class 1 (●)

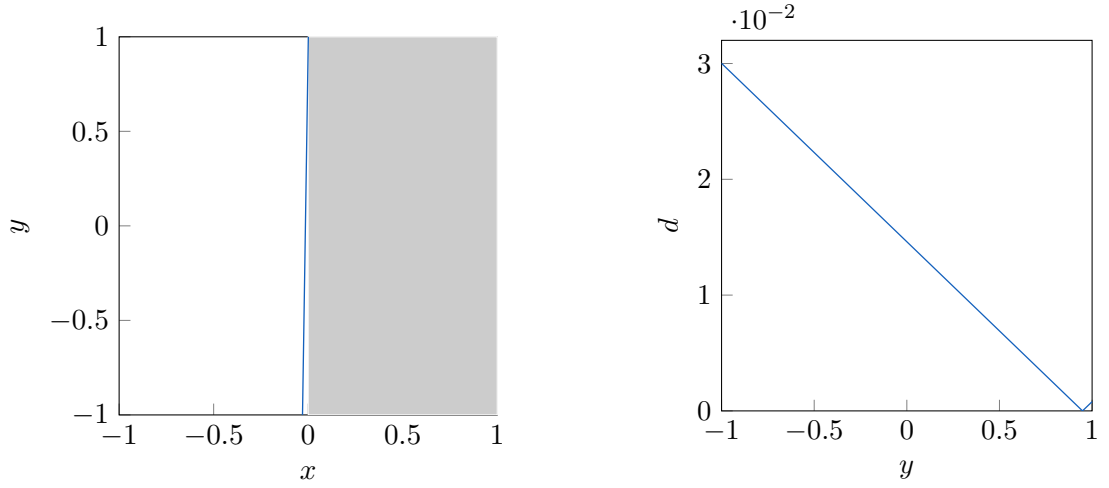
(b) Global mesh obtained with $n^0 = 6$ and $n_{LHS} = 10$ and discontinuity triangles (▲)

Figure 4.5: Potential issue of using the SVM grid for a closed discontinuity.

(b) Steep slope area

Case (b) illustrates the performances of the proposed algorithm when an area of steep slope is present. Indeed, without the introduction of jump function $L_m u$ to detect the candidates to refinement, the process would add points in the zone of steep slope, illustrating the usefulness of using both triangle's features together. For such a linear discontinuity, using a linear Kernel for the SVM classifier is more advantageous and allows reaching a better convergence of the results. Nonetheless, nonlinear features usually have polynomial shapes. This numerical parameter should be discussed depending on the application considered. In the context of blade/casing interactions, the two Kernel functions considered are the Gaussian rbf and the polynomial of the degree 3. This allows to remain general and to avoid a spline such as the Duffing oscillator jump to be approximated as linear.

The discontinuity localized with the proposed algorithm (——) is illustrated in Figure 4.6(a) and the grey and white background illustrates the actual boundary between the two smooth subdomains. The distance between both curves along y is represented in Figure 4.6(b). For these distance values, the misclassified portion δA is equal to $3 \cdot 10^{-2}$, which is larger than the prescribed termination threshold of the local refinement. Indeed, the misclassified portion is computed considering the exact discontinuity location, while the convergence criteria, used to end the local refinement process, relies on the variation between two local refinement iterations of the discontinuity location. The points evaluated during the whole process are illustrated in Figure 4.7.



(a) Localization of the discontinuity (——) obtained with the proposed methodology

(b) Distance d between the actual and the detected curves as a function of y

Figure 4.6: Localization of the discontinuity with the proposed methodology and difference d with the actual discontinuity location for case (b).

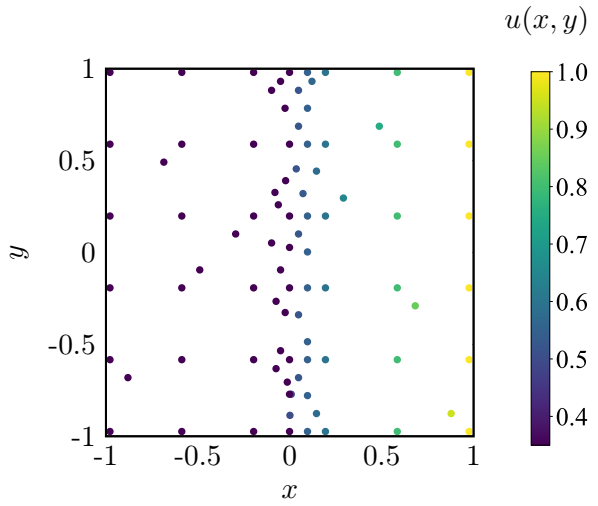


Figure 4.7: N_{total} points evaluated during the localization process for case (b).

Nonetheless, adding more points during the local refinement improves the accuracy of the jump localization. The maximum distance d_{max} between the actual and the localized curves is represented as a function of the number of points N_{local} in Figure 4.8. The error decreases when points are added. However, the accuracy is improved to some extent. When 400 points are added, the maximum error d_{max} is below $\cdot 10^{-2}$. Then lots of local points (more than 400) are required to decrease below $0.8 \cdot 10^{-2}$ as seen in Figure 4.8. The variance of the results does not reach 0 when lots of points are added, but remain between $0.1 \cdot 10^{-2}$ and $1 \cdot 10^{-2}$. This is due to numerical approximations introduced, such as the approximation of a circle by a B-spline or the Support Vector classifier score, which varies from one iteration to another.

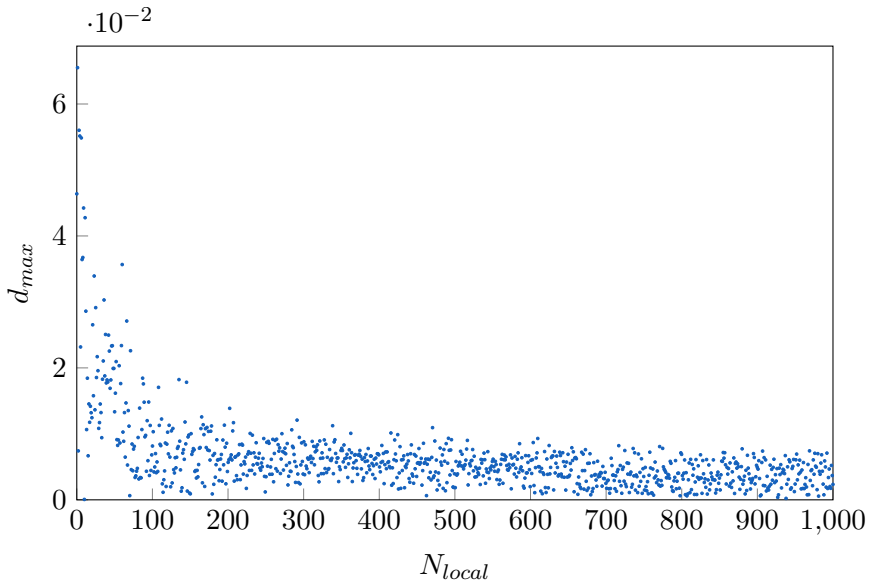


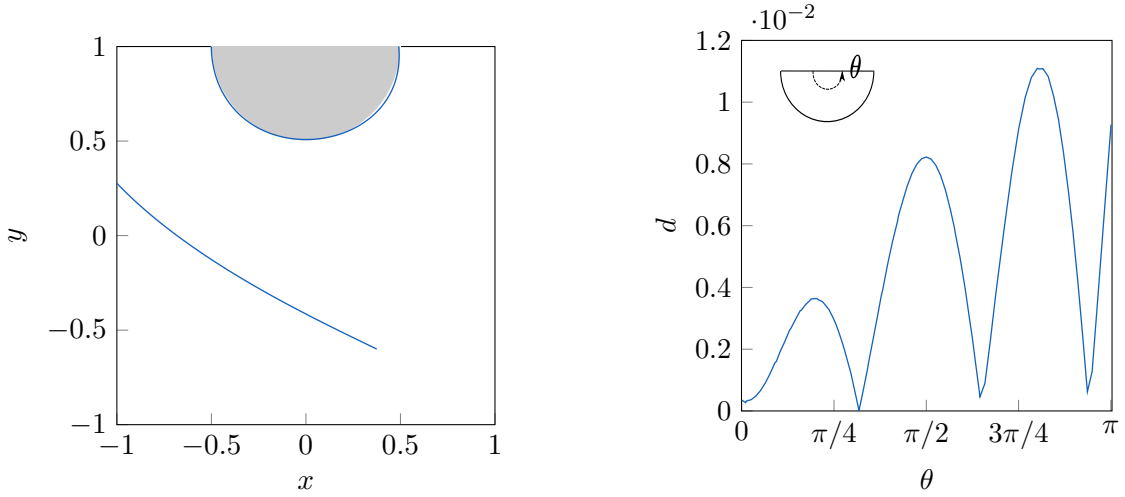
Figure 4.8: Maximum difference d_{max} between the detected and the actual discontinuity curve for an increasing number of points added during the local refinement N_{local}.

(c) Multiple discontinuities

The localization of the discontinuities for the test case (c) is illustrated in Figure 4.9(a). This case shows the performance of the algorithm when multiple discontinuities are present in the model. The grey area represents the actual boundary of the half-circle discontinuity. The distance d between both curves is computed only for the half-circle shaped discontinuity. Indeed, the Duffing jump location has no analytical expression. The distance d as a function of θ is illustrated in Figure 4.9(b). The error is below $1.2 \cdot 10^{-2}$, the same order of magnitude as the two previous cases.

As it is foreseen in the proposed method, when multiple discontinuities are present, the local refinement stopped adding points in the vicinity of the Duffing-like discontinuity, which was sufficiently refined and had continued adding points to the half-circle discontinuity. This is clearly visible in Figure 4.10, where the set of points evaluated during the whole process are represented. The density of points is more important in the vicinity of the half-circle than near the Duffing jump.

This case has shown a robustness weakness concerning the global refinement termination criteria. Indeed, using $\varepsilon = 0.1$ and the termination threshold of Table 3.2 has led simulations to add too many points in the zone without discontinuities due to the Duffing shape. As seen in Figure 4.10, numerous points have been added in the zone (⬭) $x = [0.5; 1] \times y = [-1; -0.5]$, while there is no jump there in reality. Therefore, the termination criteria, summarized in Table 3.2, could be further improved to avoid adding unnecessary points at the ends of discontinuities zones.



(a) Localization of the discontinuity (——) obtained with the proposed methodology

(b) Distance d between the actual and the detected curves as a function of θ

Figure 4.9: Localization of the discontinuity with the proposed methodology and difference d with the actual discontinuity location for case (c).

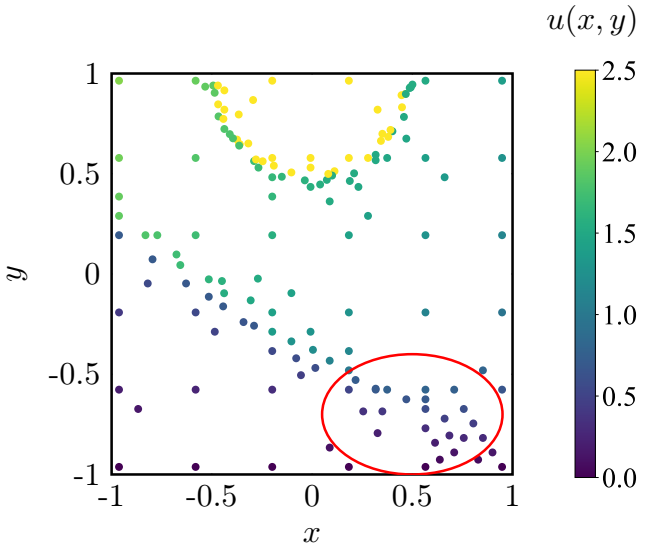
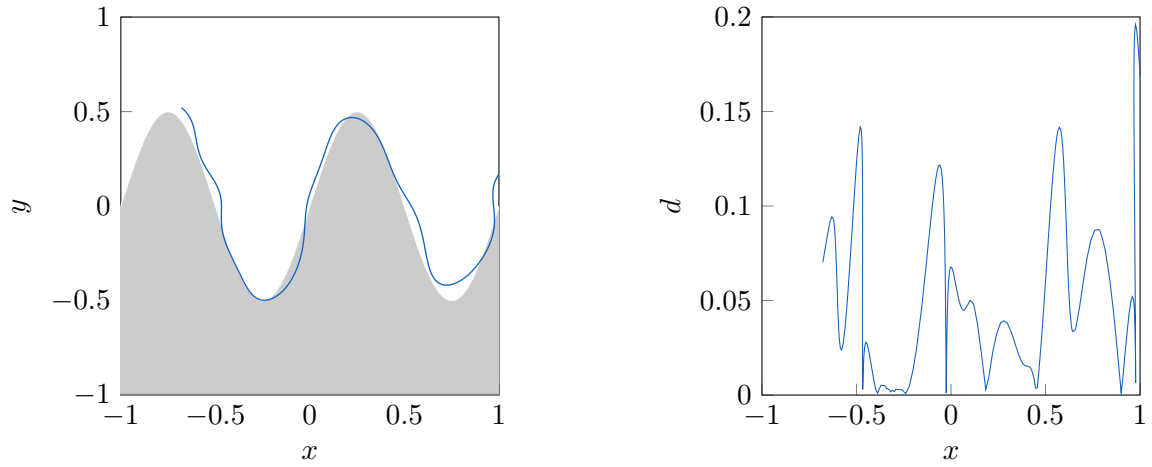


Figure 4.10: N_{total} points evaluated during the localization process for case (c).

(d) Sinusoidal shape

In the last case considered, more complex discontinuity geometry is studied along with a varying jump value. The localization of the jump (——) obtained with the proposed algorithm is illustrated in Figure 4.11(a), superimposed to a white and grey background, which delimits the boundary between the two smooth areas of the model (d). The localization obtained is less accurate than in the three previous cases studied. Indeed, the vertical distance d along x between the exact and the detected discontinuity location, represented in Figure 4.11(b), is around 10 times larger than in the previous cases. In this test case, the process has stopped adding points because the maximum number of iterations N_{max} was achieved. Therefore,



(a) Localization of the discontinuity (——) obtained with the proposed methodology

(b) Distance d between the actual and the detected curves as a function of x

Figure 4.11: Localization of the discontinuity with the proposed methodology and difference d with the actual discontinuity location for case (d).

the stabilization criteria have not been reached, and this explained the large error (~ 0.1). In addition, the left part of the curve has not been detected with the algorithm. The set of points evaluated during the whole process is illustrated in Figure 4.12.

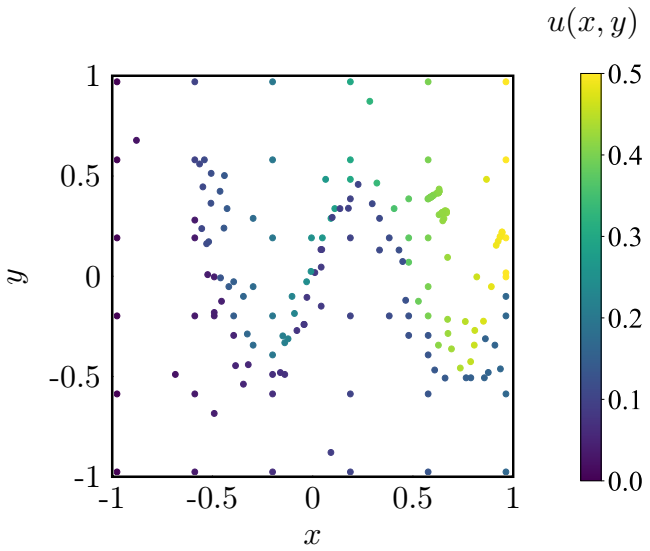


Figure 4.12: N_{total} points evaluated during the localization process for case (d).

As expected, adding more points N_{local} during the local refinement has led to a better localization of the discontinuity (——), as illustrated in Figure 4.13. This solution has required $N_{total} = 250$ evaluations points and the stabilization criteria were reached. The mean distance between the actual and the localized curves is equal to $\bar{d} = 3 \cdot 10^{-2}$.

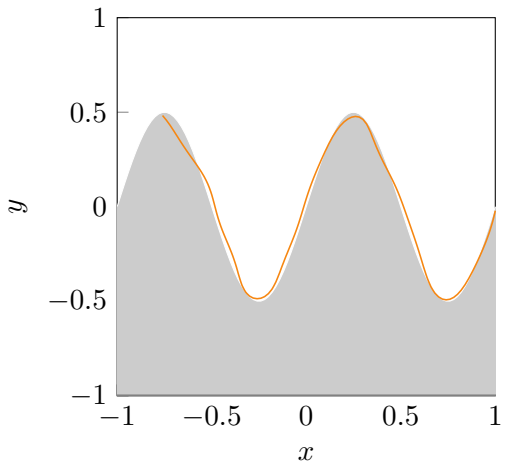


Figure 4.13: Localization of the discontinuity (——) of test case (d) obtained with a $N_{max} = 500$.

Besides, this case has shown different algorithm deficiencies. The first one concerns the global mesh. In this case, the global mesh was well constructed. Nonetheless, with other distributions of the Latin hypercube nodes, the global detection was not always properly completed. For example, two zones of discontinuities were in some cases detected instead of one. Hence, two SVM classifiers were constructed, and the discontinuity was divided into two parts.

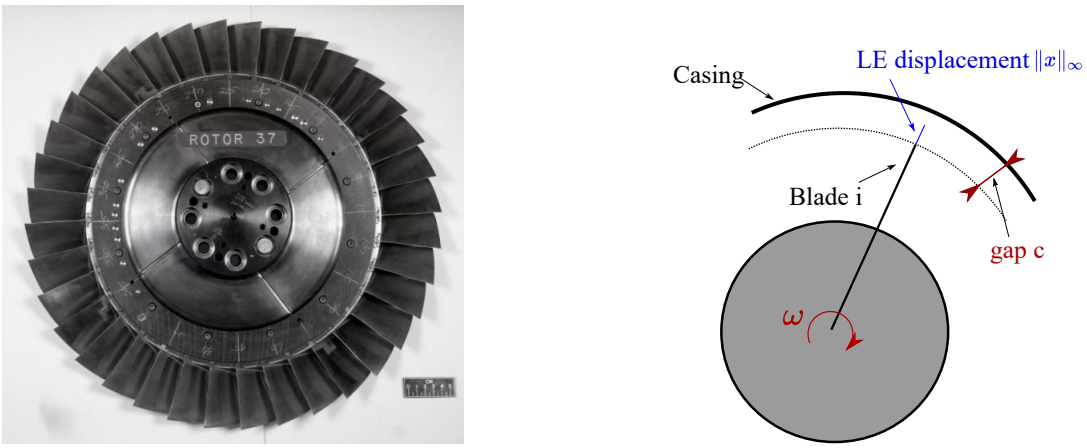
The second limitation comes from the Support Vector Classifier step. Indeed, the Support Vector Classification shows poorer results for such complex geometry. Nonetheless, as expected, increasing the number of training points during the local refinement step allows getting a better approximation of the discontinuity, but many more points are required to get a precise classifier, and the precision remains limited. As seen in Table 4.1, the parameter C was fixed to 10, illustrating the usefulness to not fix $C = 100$ when complex discontinuity geometry occurs. Indeed, using $C = 100$ could cause to overfitting which could have led to an even worst SVM classifier and discontinuity location.

Now that the algorithm has been tested on various analytical models, and that its applicability and robustness have been discussed, the proposed methodology is applied to the industrial application of interest in the context of blade/casing interactions.

4.2 Industrial application: blade/casing interactions

Now that the method has been tested on several analytical inexpensive models, the algorithm is applied in the context of blade/casing interactions. The numerical model simulates the vibratory response of a bladed disk in contact with an aircraft engine casing. The engine studied is the NASA rotor 37 illustrated in Figure 4.14(a). A finite element model has been constructed for the blade, and the vibratory behavior of the blade is simulated in the time domain using an explicit time integration scheme with contact processing by Lagrange multipliers [15]. To the best of the author's knowledge, such discontinuity detection algorithms have not been yet applied in the context of turbomachinery.

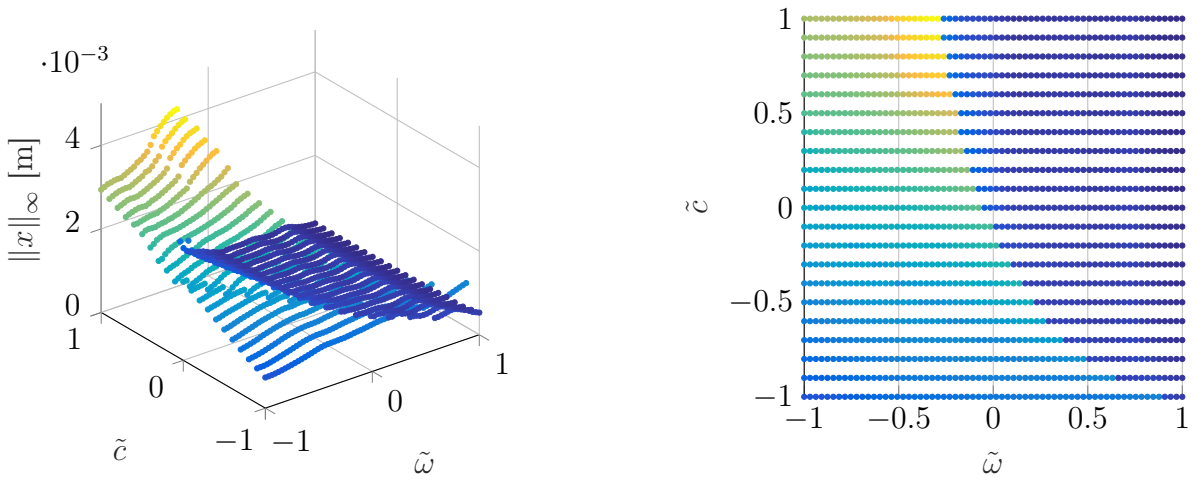
A schematic representation of the engineering model studied for the purpose of this report is illustrated in Figure 4.14(b). The two studied uncertain parameters are the size of the gap size c between the stationary and rotating parts and the frequency ω, namely the rotation speed of the engine. The infinite norm of the Leading Edge (LE) displacement $\|x\|_\infty$ of one blade is studied as a function of these two engine parameters. The gap size is assumed to vary in the range of $[2.5; 12.5]\cdot 10^{-4}$ m, and the frequency varies in the range of $[1,370; 1,500]$ rad/s. Long computation times do not allow the construction of a complete reference solution, but the response evaluated on 1,386 points is illustrated in Figure 4.15, giving already a good insight of the model of interest and the discontinuity location to be detected. To keep the same reference used for the rest of the report, the inputs are normalized in $[-1; 1]$ and writes $\tilde{c}$ and $\tilde{\omega}$.



(a) NASA rotor 37 [37]

(b) 2D schematic of the industrial model

Figure 4.14: Industrial application studied.



(a) 3D view

(b) 2D view

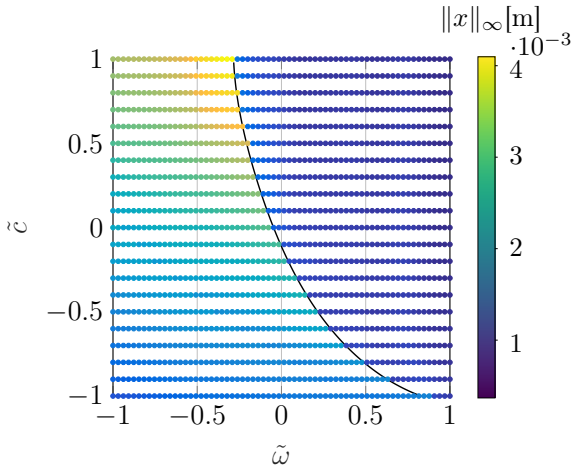
Figure 4.15: Reference solution for the blade/casing model, Leading Edge displacement as a function of the normalized clearance $\tilde{c}$ and the normalized frequency $\tilde{\omega}$.

As shown in the previous sections on the analytical test cases, the numerical parameters have to be carefully chosen. In the considered industrial application, the main problem that can occur is the global refinement process that detects two discontinuities instead of one. Nonetheless, the algorithm has shown very good results for some given basic mesh with $n_{LHS} = 10$ Latin hypercube nodes. The numerical parameters fixed for the results presented here are summarized in Table 4.3. The localization of the discontinuity (——) with the proposed algorithm is illustrated in Figure 4.16(a). The spline obtained is superimposed to the reference solution, showing that the localization of the discontinuity is highly satisfactory.

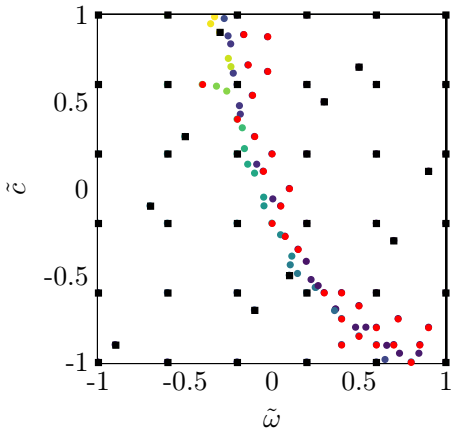
N_{max}	n^0	n_{LHS}	ε	K
200	6	10	0.1	0.1

Table 4.3: Numerical parameters used for the detection of the discontinuity for the blade/casing interactions application.

A total of 112 points have been evaluated during the whole process. The algorithm has stopped adding points since the local refinement criteria was achieved ($\delta_{x/y} < K$). All the evaluated points are illustrated in Figure 4.16(b). The basic mesh nodes (■), the points evaluated during the global refinement (●) and the points evaluated during the local refinement (●) are represented in Figure 4.16(b). The corresponding number of points for each step of the process are summarized in Table 4.4.



(a) Localization of the discontinuity (——) obtained with the proposed methodology, superimposed to the reference solution



(b) N_{total} points evaluated during the whole process

Figure 4.16: Localization of the discontinuity (——) obtained with the proposed methodology and N_{total} points evaluated during the whole process: N_{basic} (■), N_{global} (●) and N_{local} (●).

N_{basic} (■)	N_{global} (●)	N_{local} (●)	N_{total}
46	29	37	112

Table 4.4: Number of points evaluated during the different steps of the localization process for the blade/casing interactions application.

The main goal of the method is to localize the discontinuity to be able to subdivide the domain into smooth subdomains, where independent polynomial chaos expansions are applied. Therefore, the process used to localize the discontinuity should not require too much evaluations of the model. The basic mesh nodes are uniformly distributed over the whole domain. Therefore, these points are useful to construct the polynomial chaos expansion on the smooth subdomains. On the contrary, the points distributed along the discontinuity area are less useful to construct the polynomial chaos expansion. If the points distributed on the whole domain are discarded from the total number of evaluated points, a total of $N = 66$ evaluations are required to nothing other than locating the discontinuity in the Design of Experiment. The error between the actual and the detected localization of the jump in the domain cannot be computed in this case because the actual localization is not known. Nonetheless, based on the observations made with the analytical functions, one can estimate that a precision of $\sim 10^{-2}$ is achieved. The objective of the methodology, which is to localize the discontinuity as precisely as possible, with a minimum of model evaluations, has thus been achieved. Robustness improvements for the global mesh could now be made to further improve the method.

Conclusion and perspectives

The purpose of this research work was the development of an algorithm to detect and localize discontinuities in a Design of Experiments. The proposed methodology uses a Delaunay triangulation to refine the domain and get a preliminary discontinuity location. Each triangle of the mesh is characterized by two indicators based on the gradients of the model and polynomial annihilation. These two indicators allow the process to make the distinction between zones of jumps and zones of steep slopes in order to refine the mesh in the discontinuities areas. Each time a point is added and evaluated, it is assigned a label. Using the discontinuity triangles and the label of the corresponding triangles vertices, a Support Vector Classifier (SVM) is constructed to create a smooth approximation of the discontinuity location. The curve obtained through this process is then approximated as a B-spline whose control points are the output of the proposed methodology. To improve the accuracy of the discontinuity location, points are added locally in the most informative places. A new SVM classifier is then constructed, and the process is repeated until the B-spline has stabilized from one iteration to another.

The proposed methodology has been applied to four 2D analytical cases. It has proved reliable to localize closed discontinuities, multiple discontinuities in the same domain and discontinuity running partially through the domain. Between 90 to 170 evaluation points were required to localize the discontinuity in the Design of Experiments, reaching a distance between the actual and the localized spline of about $10^{-2}\%$ of the axis dimensions. Nonetheless, for more complex geometries such as a sinusoidal shape, the convergence of the results was not as good and required a lot of evaluation points. This case has highlighted some points of the method that could be further improved.

Finally, the algorithm has been tested in the context of blade/casing interactions. The model of interest was the variation of the blade tip displacement as a function of the blade tip/casing clearances and the frequency of the engine. Given a correct global refinement process, the localization of the discontinuity obtained was very satisfying.

In further research, investigating the global refinement termination threshold as a function of the number of points evaluated or as a function of the number of triangles in the domain might improve the global refinement process. Future studies could also try to change the number of enclosed points used in the polynomial annihilation method and eventually introduce the min-mod function if it is necessary at any time.

Bibliography

[1] Laboratoire d'Analyse Vibratoire et Acoustique, 2021. url:https://lava.polymtl.ca/.

[2] scikit-learn: machine learning in Python — scikit-learn 0.24.2 documentation, 2021. url:https://scikit-learn.org/stable/.

[3] SciPy v1.6.3 Reference Guide, 2021. url:https://docs.scipy.org/doc/scipy/reference/index.html.

[4] A. Abdedou, A. Soulaïmani, and G. W. Tchamen. Uncertainty propagation of dam break flow using the stochastic non-intrusive b-splines bézier elements-based method. *Journal of Hydrology*, 590:125342, 2020. doi:10.1016/j.jhydrol.2020.125342.

[5] Q. Agrapart and A. Batailly. Cubic and bicubic spline interpolation in Python. Tech.rep., École Polytechnique de Montréal, Nov. 2020. oai:hal-03017566.

[6] F. Alecu. The impact of parallel processing on operating systems. *Oeconomics of Knowledge*, 1(1):2–10, 2009. url:TheImpactOfParallel.

[7] R. Archibald, A. Gelb, R. Saxena, and D. Xiu. Discontinuity detection in multivariate space for stochastic simulations. *Computational Physics*, 228:2676–2689, 2009. doi:10.1016/j.jcp.2009.01.001.

[8] R. Archibald, A. Gelb, and J. Yoon. Polynomial fitting for edge detection in irregularly sampled signals and images. *SIAM Journal on Numerical Analysis*, 43(1):259–279, 2005. doi:10.1137/S0036142903435259.

[9] N. Bachschmid, P. Pennacchi, and A. Vania. Thermally induced vibrations due to rub in real rotors. *Journal of Sound and Vibration*, 299(4-5):683—-719, 2007. doi:10.1016/j.jsv.2006.04.045.

[10] L. Barbeau. Création de modèles de substitution pour des données présentant des failles de type saut. Tech.rep., LAVA, Aug. 2018.

[11] G. Barbujani, N. L. Oden, and R. R. Sokal. Detecting regions of abrupt change in maps of biological variables. *Systematic Zoology*, 38(4):376, 1989. doi:10.2307/2992403.

[12] M. Basu. Gaussian-based edge-detection methods-a survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 32(3):252–260, 2002. doi:10.1109/TSMCC.2002.804448 .

[13] C. M. Bishop. *Pattern Recognition and Machine Learning*, chapter 7, pages 326,327.332. Springer, 2006.

[14] C. Campbell and Y. Ying. *Learning with Support Vector Machines (Synthesis Lectures on Artificial Intelligence and Machine Learning)*. Morgan & Claypool Publishers, 1 edition, 2011.

[15] N. Carpenter, R. Taylor, and M. Katona. Lagrange constraints for transient finite element surface contact. *International Journal for Numerical Methods in Engineering*, 32(1):103–128, 1991. doi:10.1002/nme.1620320107.

[16] C.-C. Chang and C.-J. Lin. Libsvm: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.*, 2(3):27, 2011. doi:10.1145/1961189.1961199 .

[17] T. Chantrasmi, A. Doostan, and G. Iaccarino. Padé–legendre approximants for uncertainty analysis with discontinuous response surfaces. *Journal of Computational Physics*, 228(19):7159–7180, 2009. doi:10.1016/j.jcp.2009.06.024.

[18] B. Chouvion and E. Sarrouy. Development of error criteria for adaptive multi-element polynomial chaos approaches. *Mechanical Systems and Signal Processing*, 66-67:201–222, 2016. doi:10.1016/j.ymssp.2015.05.007.

[19] M. Cuny, S. Philippon, P. Chevrier, and F. Garcin. Experimental measurement of dynamic forces generated during short-duration contacts: Application to blade-casing interactions in aircraft engines. *Experimental Mechanics*, 54(2):101–114, 2014. doi:10.1007/s11340-013-9780-z.

[20] V. DalleMole, A. Araujo, R. Rego, and R. do Rego. *The Self-Organizing Approach for Surface Reconstruction from Unstructured Point Clouds*. INTECH Open Access Publisher, 2010. doi:10.5772/9180.

[21] D. Debnath, J. S. Gainer, C. Kilic, D. Kim, K. Matchev, and Y. Yang. Identifying phase-space boundaries with voronoi tessellations. *The European Physical Journal C*, 76(11):1–30, 2016. doi:10.1140/epjc/s10052-016-4431-z.

[22] D. Debnath, J. S. Gainer, D. Kim, and K. T. Matchev. Edge detecting new physics the voronoi way. *EPL (Europhysics Letters)*, 114(4):41001, 2016. doi:10.1209/0295-5075/114/410012.

[23] J. T. DeMasi-Marcin and D. K. Gupta. Protective coatings in the gas turbine engine. *Surface and Coatings Technology*, 68-69:1–9, 1994. doi:10.1016/0257-8972(94)90129-5.

[24] D. Dias-da Costa, J. Valença, E. Júlio, and H. Araújo. Crack propagation monitoring using an image deformation approach. *Structural Control and Health Monitoring*, 24(10):e1973, 2016. doi:10.1002/stc.1973.

[25] J. Dréau, B. Magnain, F. Nyssen, and A. Batailly. Polynomial chaos expansion for permutation and cyclic permutation invariant systems: Application to mistuned bladed disks. *Journal of Sound and Vibration*, 503:116103, 2021. doi:10.1016/j.jsv.2021.116103.

[26] M.-J. Fortin. Edge detection algorithms for two-dimensional ecological data. *ecology*, 75(4):956–965, 1994. doi:10.1023/A:1008194205292.

[27] M.-J. Fortin and P. Drapeau. Delineation of ecological boundaries: Comparison of approaches and significance tests. *Oikos*, 72(3):323–332, 1995. doi:org/10.2307/3546117.

[28] E. Gabriel. Détection de changements abrupts dans le gradient d'un champ gaussien et application aux sciences de l'environnement. *Journal de la société française de statistique*, 148(2):3–28, 2007. oai:tel-02833777.

[29] L. Georgep. *Génération automatique de maillages. Applications aux méthodes d'élémentsfinis*. Collection RMA, 1990.

[30] R. G. Ghanem and P. D. Spanos. *Stochastic Finite Elements: A spectral Approach*. Springer-Verlag, 1991.

[31] D. Giovanis and M. Shields. Variance-based simplex stochastic collocation with model order reduction for high-dimensional systems. *International Journal for Numerical Methods in Engineering*, 117(11):1079–1116, 2018. doi:10.1002/nme.5992.

[32] P. Gonnet. A review of error estimation in adaptive quadrature. *ACM Computing Surveys*, 44(4):1–36, 2012. doi:10.1145/2333112.2333117.

[33] A. Gorodetsky and Y. Marzouk. Efficient localization of discontinuities in complex computational simulations. *SIAM Journal on Scientific Computing*, 36(6):A2584–A2610, 2014. doi:10.1016/j.jcp.2019.04.053.

[34] H. Grabowski and X. Li. Coefficient formula and matrix of nonuniform b-spline functions. *Computer-Aided Design*, 24(12):637–642, 1992. DOI: 10.1016/0010-4485(92)90018-6.

[35] V. Gupta, S. Gupta, and J. Kim. Automated discontinuity detection and reconstruction in subsurface environment of mars using deep learning: A case study of sharad observation. *Applied Sciences*, 10:2279, 2020. doi:10.3390/app10072279.

[36] Y. V. Halder, B. Sanderse, and B. Koren. An adaptive minimum spanning tree multielement method for uncertainty quantification of smooth and discontinuous responses. *SIAM Journal on Scientific Computing*, 41(6):A3624–A3648, 2019. doi:10.1137/18m1219643.

[37] D. Hubler. Rotor 37 and stator 37 assembly. url:https://catalog.archives.gov/id/17468361.

[38] R. Iman. Latin hypercube sampling. *Encyclopedia of Quantitative Risk Analysis and Assessment*, pages 1–5, 1999. doi:10.1002/9780470061596.risk0299.

[39] G. Jacquet-Richardet, M. Torkhani, P. Cartraud, F. Thouverez, T. Nouri Baranger, M. Herran, and al. Rotor to stator contacts in turbomachines. review and application. *Mechanical Systems and Signal Processing*, 40(2):401–420, 2013. doi:10.1016/j.ymssp.2013.05.010.

[40] J. D. Jakeman, R. Archibald, and D. Xiu. Characterization of discontinuities in high-dimensional stochastic problems on adaptive sparse grids. *Journal of Computational Physics*, 230(10):3977–3997, 2011. doi:10.1016/j.jcp.2011.02.022.

[41] J. D. Jakeman, A. Narayan, and D. Xiu. Minimal multi-element stochastic collocation for uncertainty quantification of discontinuous functions. *Journal of Computational Physics*, 242:790–808, 2013. doi:10.1016/j.jcp.2013.02.035.

[42] P. Jiang, Y. Zhang, Q. Zhou, and L. Shu. A sequential sampling strategy for kriging metamodel based on delaunay triangulation and topsis. *17th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, 48(6):1644–1656, 2016. doi:10.1007/s10489-017-1031-z.

[43] G. Kewlani and K. Iagnemma. A multi-element generalized polynomial chaos approach to analysis of mobile robot dynamics under uncertainty. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1177–1182, 2009. doi:10.1109/IROS.2009.5354420.

[44] O. P. Le Maître and O. M. Knio. *Spectral Methods for Uncertainty Quantification*. Springer, 2010.

[45] O. P. Le Maître, O. M. Knio, H. N. Najm, and R. Ghanem. Uncertainty propagation using wiener–haar expansions. *Journal of Computational Physics*, 197(1):28–57, 2004. doi:10.1016/j.jcp.2003.11.033.

[46] O. P. Le Maître, H. N. Najm, R. G. Ghanem, and O. M. Knio. Multi-resolution analysis of wiener-type uncertainty propagation schemes. *Journal of Computational Physics*, 197(2):502–531, 2004. doi:10.1016/j.jcp.2003.12.020.

[47] H. Ma, F. Yin, Y. Guo, X. Tai, and B. Wen. A review on dynamic characteristics of blade–casing rubbing. *Nonlinear Dynamics*, 84(2):437—-472, 2015. doi:10.1007/s11071-015-2535-x.

[48] J. L. Meitour, D. Lucor, and J. Chassaing. Prediction of stochastic limit cycle oscillations using an adaptive polynomial chaos method. *ASD*, 2(1):3–22, 2010. doi:10.3293/ASDJ.V2I1.4.

[49] A. Millecamps, A. Batailly, M. Legrand, and F. Garcin. Snecma's viewpoint on the numerical and experimental simulation of blade-tip/casing unilateral contacts. *Structures and Dynamics*, 7B, 2015. doi:10.1115/gt2015-42682.

[50] A. Millecamps, J.-F. Brunel, P. Dufrenoy, F. Garcin, and M. Nucci. Influence of thermal effects during blade-casing contact experiments. In *ASME 2009 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, San Diego, United States, 2009. oai:hal-01223060.

[51] H. N. Najm, B. J. Debusschere, Y. M. Marzouk, S. Widmer, and O. P. Le Maître. Uncertainty quantification in chemical systems. *International Journal Numerical Methods Engineering*, 80:789–814, 2009. doi:10.1002/nme.2551.

[52] J. Noël, L. Renson, and G. Kerschen. Complex dynamics of a nonlinear aerospace structure: Experimental identification and modal interactions. *Journal of Sound and Vibration*, 333(12):2588–2607, 2014. doi:10.1016/j.jsv.2014.01.024.

[53] J. Padovan and F. K. Choy. Nonlinear Dynamics of Rotor/Blade/Casing Rub Interactions. *Journal of Turbomachinery*, 109(4):527–534, 1987. doi:10.1115/1.3262143.

[54] T. Peschoux. Plan d'expérience pour la simulation des interactions aube/carter. Tech.rep., LAVA, Feb. 2018.

[55] E. Piollet, F. Nyssen, and A. Batailly. Blade/casing rubbing interactions in aircraft engines: Numerical benchmark and design guidelines based on nasa rotor 37. *Journal of Sound and Vibration*, 460:114878, 2019. doi:10.1016/j.jsv.2019.114878.

[56] H. Prautzsch, W. Boehm, and M. Paluszny. *Bézier representation*. Springer Berlin Heidelberg, 2002. doi:10.1007/978-3-662-04919-8_2.

[57] R. Rajasekharan and E. P. Petrov. Analysis of Deformation of Mistuned Bladed Disks With Friction and Random Crystal Anisotropy Orientation Using Gradient-Based Polynomial Chaos Expansion. *J. Eng. Gas Turbines Power.*, 141(4), 2019. doi:10.1115/1.4040906.

[58] L. Reid and R. Moore. Design and overall performance of four highly loaded, high-speed inlet stages for and advanced high-pressure-ratio core compressor. Tech.rep. NASA TP-1337, NASA Lewis Research Center, 1978. url:https://ntrs.nasa.gov/citations/19780025165.

[59] B. Rosner. Percentage Points for a Generalize ESD Many-Outlier Procedure. *Technometrics*, 25(2):165–172, 1983. doi:10.2307/1268549.

[60] A. A. Rushdi, L. P. Swiler, E. T. Phipps, M. D'Elia, and M. Ebeida. Voronoi piecewise surrogate models for high-dimensional data fitting. *International Journal for Uncertainty Quantification*, 7(1):1–21, 2017. doi:10.1615/Int.J.UncertaintyQuantification.2016018697.

[61] K. Sargsyan, C. Safta, B. Debusschere, and H. Najm. Uncertainty quantification given discontinuous model response and a limited number of model runs. *SIAM Journal on Scientific Computing*, 34(1):B44–B64, 2012. doi:10.1137/100817899.

[62] D. Schiavazzi, A. Doostan, G. Iaccarino, and A. Marsden. A generalized multi-resolution expansion for uncertainty propagation with application to cardiovascular modeling. *Computer Methods in Applied Mechanics and Engineering*, 314:196–221, 2017. doi:10.1016/j.cma.2016.09.024.

[63] N. C. Schwertman and R. de Silva. Identifying outliers with sequential fences. *Computational Statistics & Data Analysis*, 51(8):3800–3810, 2007. doi:10.1016/j.csda.2006.01.019.

[64] S. Shahane, N. R. Aluru, and S. P. Vanka. Uncertainty quantification in three dimensional natural convection using polynomial chaos expansion and deep neural networks. *International Journal of Heat and Mass Transfer*, 139:613–631, 2019. doi:10.1016/j.ijheatmasstransfer.2019.05.014.

[65] M. Shepperd, D. Bowes, and T. Hall. Researcher bias: The use of machine learning in software defect prediction. *IEEE Transactions on Software Engineering*, 40(6):603–616, 2014. doi:10.1109/TSE.2014.2322358.

[66] M. Spiridonakos and E. Chatzi. Metamodeling of dynamic nonlinear structural systems through polynomial chaos narx models. *Computers & Structures*, 157:99–113, 2015. doi:10.1016/j.compstruc.2015.05.002.

[67] I. Steinwart and C. Scovel. Mercer's theorem on general domains: On the interaction between measures, kernels, and rkhss. *Constructive Approximation*, 35(3):363–417, 2012. DOI: 10.1007/s00365-012-9153-3 .

[68] P. Thanh Noi and M. Kappas. Comparison of Random Forest, k-Nearest Neighbor, and Support Vector Machine Classifiers for Land Cover Classification Using Sentinel-2 Imagery. *Sensors*, 18(2):18, 2017. doi:10.3390/s18010018.

[69] A. Ukil and R. Živanović. Abrupt change detection in power system fault analysis using adaptive whitening filter and wavelet transform. *Electric Power Systems Research*, 76(9):815–823, 2006. doi:10.1016/j.epsr.2005.10.009.

[70] D. van Ravenzwaaij, P. family=Cassey, and S. Brown. A simple introduction to markov chain monte–carlo sampling. *Psychonomic Bulletin & Review*, 25(1):143–154, 2016. doi:10.3758/s13423-016-1015-8.

[71] E. Walter. Méthodes numériques et optimisation, un guide du consommateur. *ffhal-01238558*, page 80, 2015. oai:01238558.

[72] X. Wan and G. E. Karniadakis. An adaptive multi-element generalized polynomial chaos method for stochastic differential equations. *Journal of Computational Physics*, 209(2):617–642, 2005. doi:10.1016/J.JCP.2005.03.023.

[73] A. Widodo and B.-S. Yang. Support vector machine in machine condition monitoring and fault diagnosis. *Mechanical Systems and Signal Processing*, 21(6):2560—-2574, 2007. doi:10.1016/j.ymssp.2006.12.007.

[74] T. Wildey, A. A. Gorodetsky, A. Belme, and J. N. Shadid. Robust uncertainty quantification using response surface approximations of discontinuous functions. *International Journal for Uncertainty Quantification*, 9(5):415–437, 2019. doi:10.1615/Int.J.UncertaintyQuantification.2019026974.

[75] J. A. Witteveen, S. Sarkar, and H. Bijl. Modeling physical uncertainties in dynamic stall induced fluid–structure interaction of turbine blades using arbitrary polynomial chaos. *Computers & Structures*, 85(11):866–878, 2007. doi:10.1016/j.compstruc.2007.01.004.

[76] W. H. Womble. Differential systematics. *Science*, 114(2961):315–322, 1951. doi:10.1126/science.114.2961.315.