ETH zürich

A TWO-STAGE SURROGATE MODELLING APPROACH FOR THE APPROXIMATION OF MODELS WITH NON-SMOOTH OUTPUTS

M. Moustapha and B. Sudret

CHAIR OF RISK, SAFETY AND UNCERTAINTY QUANTIFICATION
STEFANO-FRANSCINI-PLATZ 5
CH-8093 ZÜRICH

Risk, Safety & Uncertainty Quantification

Data Sheet

Journal:	3rd ECCOMAS Thematic Conference on Uncertainty Quantification in Computational Sciences and Engineering
Report Ref.:	RSUQ-2019-009
Arxiv Ref.:	-
DOI:	-
Date submitted:	March 31st, 2019
Date accepted:	June 24, 2019

UNCECOMP 2019
3rd ECCOMAS Thematic Conference on
Uncertainty Quantification in Computational Sciences and Engineering
M. Papadrakakis, V. Papadopoulos, G. Stefanou (eds.)
Crete, Greece, 24-26 June 2019

A TWO-STAGE SURROGATE MODELLING APPROACH FOR THE APPROXIMATION OF MODELS WITH NON-SMOOTH OUTPUTS

Maliki Moustapha[1], **Bruno Sudret**[1]

[1]Chair of Risk, Safety and Uncertainty Quantification
ETH Zurich
e-mail: {moustapha,sudret}@ibk.baug.ethz.ch

Keywords: Surrogate modelling, Non-smooth outputs, Support vector machines, Kriging.

Abstract. *Surrogate modelling has become an important topic in the field of uncertainty quantification as it allows for the solution of otherwise computationally intractable problems. The basic idea in surrogate modelling consists in replacing an expensive-to-evaluate black-box function by a cheap proxy. Various surrogate modelling techniques have been developed in the past decade. They always assume accommodating properties of the underlying model such as regularity and smoothness. However such assumptions may not hold for some models in civil or mechanical engineering applications, e.g., due to the presence of snap-through instability patterns or bifurcations in the physical behavior of the system under interest. In such cases, building a single surrogate that accounts for all possible model scenarios leads to poor prediction capability. To overcome such a hurdle, this paper investigates an approach where the surrogate model is built in two stages. In the first stage, the different behaviors of the system are identified using either expert knowledge or unsupervised learning, i.e. clustering. Then a classifier of such behaviors is built, using support vector machines. In the second stage, a regression-based surrogate model is built for each of the identified classes of behaviors. For any new point, the prediction is therefore made in two stages: first predicting the class and then estimating the response using an appropriate recombination of the surrogate models. The approach is validated on two examples, showing its effectiveness with respect to using a single surrogate model in the entire space.*

1 INTRODUCTION

The surrogate modelling of computer simulations has become paramount in many engineering applications that rely heavily on high-fidelity models. Surrogate models indeed allow for an inexpensive approximation of the model input-output relationship, thus making computationally intensive analyses, such as design optimization or uncertainty quantification, affordable. In the common setting, the underlying surrogated model is assumed to exhibit accommodating properties such as smoothness and continuity. However, numerous engineering problems involve non-smooth functions, e.g. crash simulation in the automotive industry. Indeed the original model may exhibit some sharp localized features and discontinuities may occur when a bifurcation or an instability appears in the solution path. In general, the functions of interest in this work exhibit different behaviors which can be mapped to certain combinations of the input parameters. The transitions between these domains may be non-smooth, often featuring discontinuities. In mechanical engineering, typical examples are buckling and snap-through characterized by sudden behavior changes (See [3] for instance). Approximating such models with a traditional smooth surrogate model leads to large errors. In this work, we consider a two-stage approach where the different behaviors are first localized and classified and then locally approximated. A similar approach was investigated in [1] and [8], albeit without approximation of the model responses as the latter were used as constraints in an optimization setting where only feasibility of a given design is of interest. In this paper, the general workflow of the proposed methodology is first introduced. This is followed by a brief description of the different blocks of the algorithm. Finally, two application examples are used to show the effectiveness of the proposed approach.

2 PROPOSED APPROACH

2.1 Workflow of the method

The proposed approach for handling non-smooth functions consists of multiple steps as described in the flowchart of Figure 1. Let us consider an experimental design which consists of N uniformly sampled points $\left\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(N)}\right\}$ and their corresponding model evaluations $\left\{y^{(1)} = \mathcal{M}\left(\boldsymbol{x}^{(1)}\right), \ldots, y^{(N)} = \mathcal{M}\left(\boldsymbol{x}^{(N)}\right)\right\}$. To build the predictor, the following steps are undertaken:

1. *Clustering*: This is the first step of the approach when the analyst attributes to each observation $\boldsymbol{y}^{(i)}, i = \{1, \ldots, N\}$ a class which corresponds to an identified behavior of the system. In the ideal case, this can be done manually using expert knowledge. In the general case though, it is more convenient to rely on an automated approach where the classes are directly learned from the data using *unsupervised learning*.

2. *Classification*: Once the classes are clearly identified, they are mapped to the input space which is then partitioned accordingly. This step is carried out using support vector machines for classification as detailed in the next section.

3. *Regression/Interpolation*: Eventually, the dataset is split into the different groups identified in the previous two steps. For each group, a local surrogate model $\left\{\widehat{\mathcal{M}}_k,\, k = 1, \ldots, K\right\}$ is built.

Once the local models are built, it is necessary to recombine them when evaluating a new point. As shown on the right side of Figure 1, this is achieved in three steps:

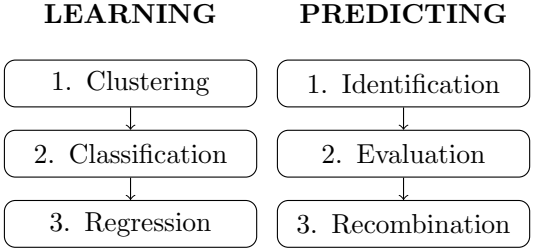


Figure 1: Illustration of the surrogate modelling approach.

1. *Identification*: The very first step is to predict to which class belongs the new point. The previously built support vector classifier can be used in that respect.
2. *Evaluation*: The new point is then evaluated using the different surrogate models.
3. *Recombination*: The final approximation is obtained by combining the different predictions as follows:

$$\widehat{\mathcal{M}}\left(\boldsymbol{x}\right) = \sum_{k=1}^{K} w_k\left(\boldsymbol{x}\right) \widehat{\mathcal{M}}_k\left(\boldsymbol{x}\right), \tag{1}$$

$w_k\left(\boldsymbol{x}\right)$ are weight functions defined such that $\sum_{k=1}^{K} w_k\left(\boldsymbol{x}\right) = 1$. Two different types of weight functions are considered in this work as explained in the next section. In the sequel, we first describe briefly the two surrogate model types used here, namely support vector machines and Kriging.

2.2 Support vector machines for classification basics

Support vector machines are a powerful learning technique developed by Vapnik ([11]) for classification (SVC) and regression (SVR) problems. Let us consider a dataset $\mathcal{C} = \left\{\left(\boldsymbol{x}^{(i)}, \ell^{(i)}\right), i = 1, \ldots, N\right\}$, where $\boldsymbol{x}^{(i)}$ are M-dimensional input points and $\ell^{(i)} = \{-1, 1\}$ are the corresponding labels, in the particular case of binary classification considered here.

The support vector classifier is a function of the following form ([9]):

$$\mathcal{M}^{\text{SVC}}\left(\boldsymbol{x}\right) = \sum_{i=1}^{N} \alpha_i\, \ell^{(i)}\, k\left(\boldsymbol{x}^{(i)}, \boldsymbol{x}\right) + b, \tag{2}$$

where α_i and b are coefficients to calibrate and $k\left(\right)$ is the so-called *kernel* function. The coefficients of the expansion are actually found by solving the following optimization problem ([9]):

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \frac{1}{2}\boldsymbol{\alpha}^T\left(\widetilde{\boldsymbol{K}}\boldsymbol{Y}\boldsymbol{Y}^T\right)\boldsymbol{\alpha} + \boldsymbol{c}^T\boldsymbol{\alpha} \\ \text{subject to:} \quad & \boldsymbol{\alpha}^T\boldsymbol{Y} = 0, \qquad \alpha_i \geq 0, \qquad i = \{1, \ldots, N\}, \end{aligned} \tag{3}$$

where $\boldsymbol{c} = \{-1, \ldots, -1\}$ is a column vector of size N and $\widetilde{\boldsymbol{K}} = \boldsymbol{K} + 1/C\boldsymbol{I}_N$. In the latter equation, $\boldsymbol{K}$ is the so-called Gram matrix whose components read $K_{ij} = k\left(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)}\right)$ for $i, j \in \{1, \ldots, N\}$, $\boldsymbol{I}_N$ is the identity matrix of size N and C is a penalty coefficient which acts as a regularization term against overfitting.

In this work, we consider the Matérn $5/2$ kernel which reads:

$$k\left(\boldsymbol{x}, \boldsymbol{x}'\right) = \left(1 + \sqrt{5}\frac{\|\boldsymbol{x} - \boldsymbol{x}'\|}{\gamma} + \frac{5}{3}\frac{\|\boldsymbol{x} - \boldsymbol{x}'\|^2}{\gamma^2}\right)\exp\left(-\sqrt{5}\frac{\|\boldsymbol{x} - \boldsymbol{x}'\|}{\gamma}\right), \tag{4}$$

where $\gamma > 0$ is a parameter that needs to be calibrated. Together with the penalty term C, they constitute the set of hyperparameters $\boldsymbol{\theta} = \{C, \gamma\}$ whose proper calibration is crucial for the accuracy of the trained SVC model. Here they are calibrated by minimizing the span estimate of the leave-one-out error ([10]).

After setting up the model, the predicted boundary between the two classes is defined by $\left\{\boldsymbol{x} \in \mathbb{X} : \mathcal{M}^{\text{SVC}}(\boldsymbol{x}) = 0\right\}$ while the class of a prediction is given by $\text{sign}\left(\mathcal{M}^{\text{SVC}}(\boldsymbol{x})\right)$.

2.3 Kriging basics

Kriging a.k.a. Gaussian process modelling is a surrogate modelling technique where the function to approximate is considered to be the realization of a stochastic Gaussian process which reads ([6, 7]):

$$\mathcal{M}(\boldsymbol{x}) = \sum_{j=1}^{p} \beta_j f_j(\boldsymbol{x}) + Z(\boldsymbol{x}), \tag{5}$$

where f_j and β_j are a set of p regressors and their corresponding coefficients and $Z(\boldsymbol{x})$ is a second-order zero-mean stationary Gaussian process whose covariance reads $\text{Cov}[\boldsymbol{x}, \boldsymbol{x}'] = \sigma^2 R(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\gamma})$. In the latter equation, σ^2 is a constant variance of the process and R is an auto-correlation function with parameters $\boldsymbol{\gamma}$.

The auto-correlation function encodes assumptions made about the function to approximate, *e.g.* smoothness, derivability, etc. In this work, we consider the Matérn $5/2$ auto-correlation as in Eq. (4). The training of the Kriging model is a two-step process. First, the coefficients of the regression together with the process variance are estimates using least-square or equivalently maximum likelihood minimization. Second, the optimal parameters of the auto-correlation function are estimated using cross-validation or maximum likelihood. Once the estimates of the hyperparameters $\left\{\widehat{\boldsymbol{\beta}}, \widehat{\boldsymbol{\sigma}}^2, \widehat{\boldsymbol{\gamma}}\right\}$ are set, the prediction for a new point is assumed to follow a Gaussian distribution whose mean is the actual Kriging predictor and reads:

$$\mu_{\mathcal{M}}(\boldsymbol{x}) = \boldsymbol{f}^T(\boldsymbol{x})\,\widehat{\boldsymbol{\beta}} + \boldsymbol{r}^T(\boldsymbol{x})\,\boldsymbol{R}^{-1}\left(\boldsymbol{y} - \boldsymbol{F}^T\widehat{\boldsymbol{\beta}}\right), \tag{6}$$

where $\boldsymbol{R}$ is the Gram matrix defined such that $R_{ij} = R\left(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)}; \widehat{\gamma}\right)$, $\boldsymbol{r}(\boldsymbol{x}) = \left\{R\left(\boldsymbol{x}, \boldsymbol{x}^{(i)}; \widehat{\gamma}\right), i = 1, \ldots, N\right\}$ is a cross-correlation vector, $\boldsymbol{F} = \left\{F_{ij} = f_j\left(\boldsymbol{x}^{(i)}\right), i = 1, \ldots, N, j = 1, \ldots, p\right\}$ and $\boldsymbol{y} = \left\{y_i = \mathcal{M}\left(\boldsymbol{x}^{(i)}\right), i = 1, \ldots, N\right\}$ are the observations in the experimental design.

2.4 Models recombination using SVC

In the second step of the approach, an SVC model is used to partition the space. In this paper, only cases with two possible behavior scenarios are considered. Let us assume now that the two sub-regions of the space corresponding to the negative and positive labels of the classifiers are respectively denoted by $\mathcal{R}_1$ and $\mathcal{R}_2$. As explained above, the experimental design $\mathcal{D}$ is split in two subsets $\mathcal{D}_k = \left\{\left(\boldsymbol{x}^{(i)}, y^{(i)}\right) \in \mathcal{D} : \boldsymbol{x}^{(i)} \in \mathcal{R}_k\right\}, k = \{1, 2\}$. Using the subset $\mathcal{D}_1$ (resp. $\mathcal{D}_2$), a Kriging model denoted by $\widehat{\mathcal{M}_1}$ (resp. $\widehat{\mathcal{M}_2}$) is built.

Let us now consider a new point $\boldsymbol{x}$ to evaluate. As described above, we first predict its class, $\text{sign}\left(\mathcal{M}^{\text{SVC}}(\boldsymbol{x})\right)$, using the classifier. This point is then evaluated using the local surrogate models which are eventually recombined following Eq. (1). Two recombination schemes are considered here:

Binary approach

In this case, only the model built over the region in which $\boldsymbol{x}$ is predicted to belong to is used ([2, 4]). The weight function is therefore a simple indicator function, *i.e.* :

$$w_k(\boldsymbol{x}) = \mathbb{1}_{\mathcal{R}_k}(\boldsymbol{x}) = \begin{cases} 1 & \text{if } \boldsymbol{x} \in \mathcal{R}_k, \\ 0 & \text{otherwise,} \end{cases} \tag{7}$$

For the case with only two possible scenarios considered here, Eq. (1) can then be simplified into:

$$\widehat{\mathcal{M}}(\boldsymbol{x}) = \mathbb{1}_{\mathcal{R}_1}(\boldsymbol{x})\,\widehat{\mathcal{M}}_1(\boldsymbol{x}) + \mathbb{1}_{\mathcal{R}_2}(\boldsymbol{x})\,\widehat{\mathcal{M}}_2(\boldsymbol{x}). \tag{8}$$

This is a simple approach but it may yield large errors when the classification of the new point is wrong. The next approach tackles this issue by considering the uncertainty related to the support vector machine classifier.

Weighting approach

In this case, weights associated to each model are computed using the SVC prediction. The more likely a point is to belong to a class, the higher the corresponding weight and vice-versa. To compute the weight, the output of the classifier is post-processed into posterior probabilities using the following parametric sigmoid ([5]):

$$\mathbb{P}\left(\ell(\boldsymbol{x}) = 1 | \mathcal{M}^{\text{SVC}}(\boldsymbol{x})\right) = \frac{1}{1 + \exp\left(A\,\mathcal{M}^{\text{SVC}}(\boldsymbol{x}) + B\right)}, \tag{9}$$

where A and B are parameters that are fit using maximum likelihood estimation on the experimental design. The final prediction is then obtained by setting these probabilities as weights, *i.e.* :

$$w_1(\boldsymbol{x}) = 1 - \mathbb{P}\left(\ell(\boldsymbol{x}) = 1 | \mathcal{M}^{\text{SVC}}(\boldsymbol{x})\right) \quad \text{and} \quad w_2(\boldsymbol{x}) = \mathbb{P}\left(\ell(\boldsymbol{x}) = 1 | \mathcal{M}^{\text{SVC}}(\boldsymbol{x})\right). \tag{10}$$

3 APPLICATIONS

We consider two applications to illustrate the proposed methodology, namely a two-dimensional mathematical function and a snap-though mechanical problem.

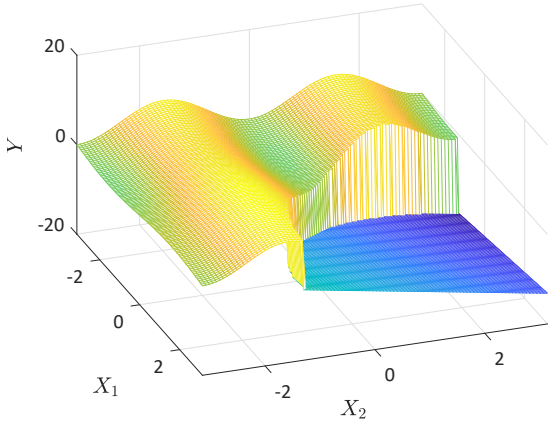
3.1 Two-dimensional mathematical function

Let us consider the two-dimensional mathematical function defined by:

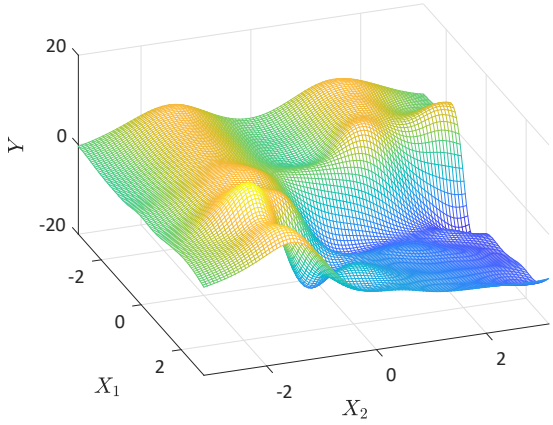
$$\mathcal{M}(\boldsymbol{x}) = \begin{cases} \sin(x_1) + 7\sin(x_2)^2 & \text{if } (x_1 - \pi)^2 + (x_2 - \pi)^2 - 2\pi^2 \geq 0, \\ x_1 - 2x_2 - 10; & \text{otherwise,} \end{cases} \tag{11}$$

where $\boldsymbol{x} \in [-\pi, \pi]^2$.

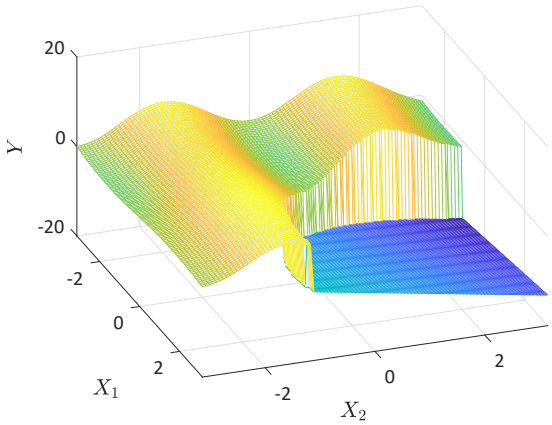
Figure 2a illustrates the function which consists of two distinct regions over which different behaviors of the model can be observed. On one side, the function is highly non-linear whereas on the other, the function is linear and nearly flat. To approximate this function, we use an experimental design of size 100. The two classes are identified using K-means clustering and the input space is partitioned as illustrated in Figure 3 by support vector machines. In this figure, the training points that belong to the flat and highly non-linear regions are shown in blue circles and red squares respectively. With a 100-point training set, the classifier, shown by the black curve, is close enough to the true one, shown by the magenta curve. After building surrogates in



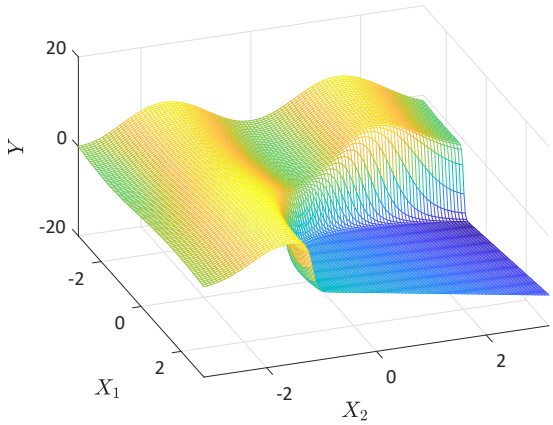
(a) Original model



(b) Approximation: One single model



(c) Approximation: Binary recombination



(d) Approximation: Weighted recombination

Figure 2: Two-dimensional mathematical problem: original *vs.* surrogate models

each region, the two recombination schemes are applied. Figures 2c and 2d show the resulting approximations. The binary case produces a very accurate representation of the model, the only error being the position of the discontinuity. The weighted recombination scheme produces a smooth transition in the margin between the two regions. Finally, using one single surrogate model leads to the approximation shown in Figure 2b where spurious curvatures are added in the vicinity of the discontinuity.

For a quantitative comparison of the different approaches, the following two errors metrics are considered:

$$\begin{aligned} NMSE &= \sum_{i=1}^{N_{val}} \left(\mathcal{Y}_i - \widehat{\mathcal{Y}}_i\right)^2 / \sum_{i=1}^{N_{val}} \left(\mathcal{Y}_i - \bar{\mathcal{Y}}\right)^2, \\ MAE &= \sum_{i=1}^{N_{val}} \left|\mathcal{Y}_i - \widehat{\mathcal{Y}}_i\right| / N, \end{aligned} \tag{12}$$

where $NMSE$ and MAE respectively stand for *normalized mean square error* and *mean absolute error*. In these equations, $\mathcal{Y}$ and $\widehat{\mathcal{Y}}$ are responses of the original and surrogate models on a validation set of size $N_{val} = 10,000$. Table 1 shows the resulting errors where cases $\#1$, $\#2$ and $\#3$ respectively stand for single surrogate, binary recombination and weighted recombina-

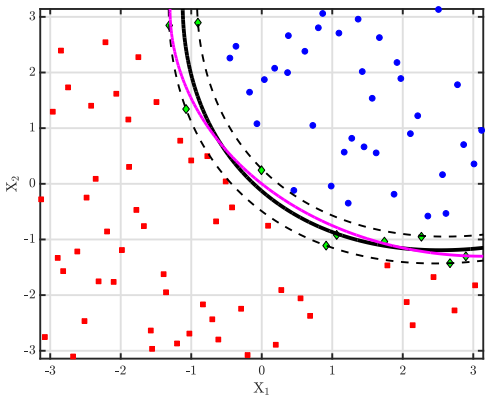


Figure 3: Two-dimensional mathematical problem: classification of the input points using support vector machines

tion. The proposed approach improves the prediction considering any of the two metrics. It is not clear though which of the two recombination schemes is more effective.

	Case #1	Case #2	Case #3
$NMSE$	0.0911	0.0530	0.0346
MAE	1.0124	0.2048	0.2436

Table 1: Two-dimensional mathematical problem: comparison of the resulting errors

3.2 Truss structure subject to snap-through

The second example addresses the problem of a geometrically non-linear two-bar truss structure with a snap-through behavior as illustrated in Figure 4. When loaded, such a structure often behaves linearly with small displacements. However, when a critical limit is reached, the structure becomes unstable and undergoes a sudden large displacement by snapping through another equilibrium point. In this example, we approximate the displacements w of the tip of such a structure considering the random parameters shown in Table 2.

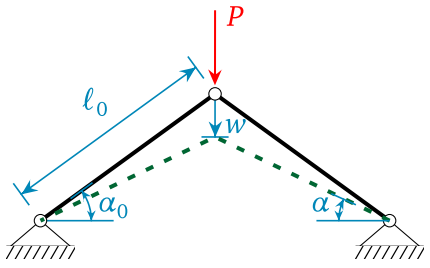


Figure 4: Illustration of the truss structure subject to snap-through

It can be shown that the load at a deformed position follows a relationship given by:

$$P = -2EA\tan(\alpha)\left(\cos(\alpha_0) - \cos(\alpha)\right) \tag{13}$$

Parameter	Distribution	Mean	C.o.V.
Load (P in N)	Gumbel	430	0.20
Young's modulus (E in GPa)	Lognormal	210	0.10
Cross sectional area (A in cm^2)	Gaussian	10	0.05

Table 2: Truss snap-through problem: probabilistic input model

where α_0 and α are the inclination angles of the bars at the initial and deformed positions. The corresponding displacement of the tip of the truss then reads:

$$w = l_0 \cos(\alpha_0) (\tan(\alpha_0) - \tan(\alpha)) . \tag{14}$$

In this example, we set $l_0 = 5$ m and $\alpha_0 = 10°$. Using an experimental design of 100 points drawn following the distribution in Table 2, the displacements are computed and shown in Figure 5. We can clearly observe the two behaviors that lead to entirely different displacements.

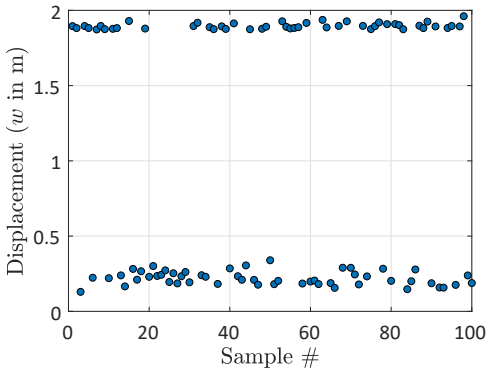
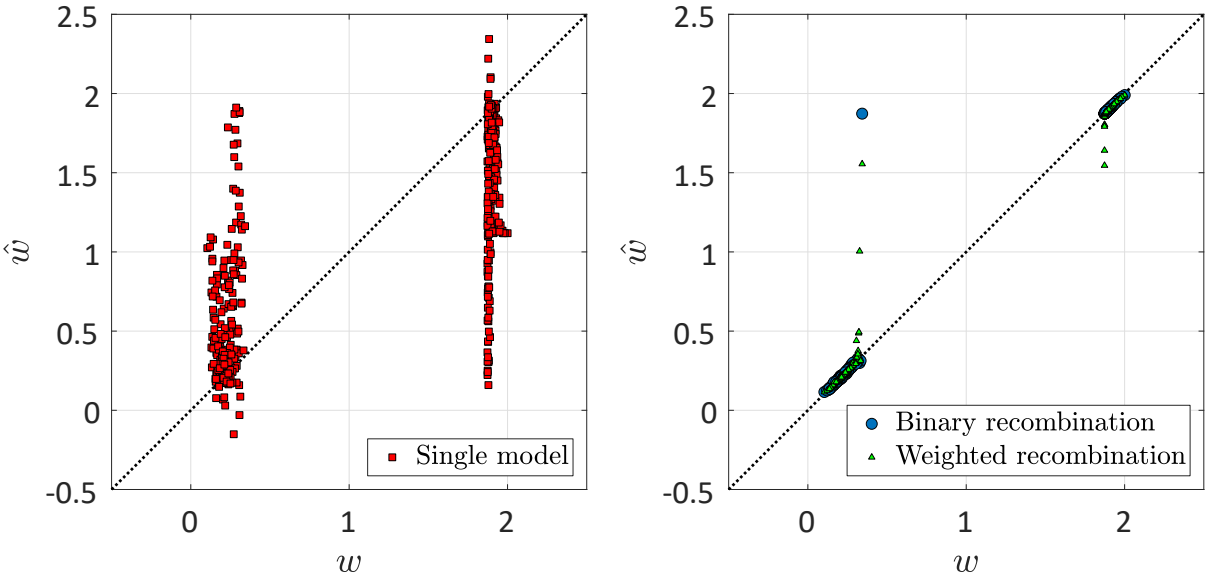


Figure 5: Truss snap-through problem: experimental design model responses

The proposed approach is applied to this experimental design. Figure 6 and Table 3 show the results for comparison. When using a single surrogate model, the instability is not captured and displacements are predicted continuously over the two extreme cases. The proposed approach allows to accurately locate and isolate the input sub-regions that lead to each of the scenarios. The binary approach produces extremely accurate results as long as the class is correctly predicted by the SVM model. The weighted recombination scheme yields locally less accurate results but behaves better than the binary one close to the discontinuity.

	Case #1	Case #2	Case #3
$NMSE$	0.2478	0.0803	0.0670
MAE	0.2714	0.0390	0.0399

Table 3: Truss snap-through problem: comparison of the resulting errors



(a) One single model

(b) Multiple models recombined

Figure 6: Truss snap-through problem: original *vs.* predicted responses

4 CONCLUSION

This paper presents a two-stage approach for the approximation of functions with non-smooth outputs. Focus is given to the particular case when multiple behaviors of the function can be observed. The proposed approach consists in first identifying such behaviors and then classifying them using support vector machines. The resulting prediction is obtained by building local surrogates in each region and then recombining them using two different schemes. Two application examples show the efficiency of the approach with respect to using a unique global surrogate model. The accuracy of the resulting predictions however relies on the accuracy of the classification step. The latter can be increased by using adaptive sampling scheme in order to more accurately define the boundaries between the two regions. Furthermore, the proposed scheme is limited to binary problems and will be extended to the more general case when more than two behaviors of the system can be observed.

References

[1] Basudhar, A., S. Missoum, and A. H. Sanchez (2008). Limit state function identification using Support Vector Machines for discontinuous responses and disjoint failure domains. *Prob. Eng. Mech 23*, 1–11.

[2] Boroson, E. and S. Missoum (2017). Stochastic optimization of nonlinear energy sinks. *Struct. Multidisc. Optim. 55*, 633–646.

[3] Hrinda, G. A. (2010). Snap-through instability patterns in truss structures. In *Proc. 51st AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, 12 - 15 April 2010, Orlando, Florida*.

[4] Moustapha, M. (2016). *Adaptive surrogate models for the reliable lightweight design of automotive body structures*. Ph. D. thesis, Université Blaise Pascal, Clermont-Ferrand, France.

[5] Platt, J. C. (1999). Advances in kernel methods. Chapter Fast training of support vector machines using sequential minimal optimization, pp. 185–208. MIT Press.

[6] Rasmussen, C. E. and C. K. I. Williams (2006). *Gaussian processes for machine learning* (Internet ed.). Adaptive computation and machine learning. Cambridge, Massachusetts: MIT Press.

[7] Santner, T. J., B. J. Williams, and W. I. Notz (2003). *The Design and Analysis of Computer Experiments*. Springer, New York.

[8] Serna, A. and C. Bucher (2009). Advanced surrogate models for multidisciplinary design optimization. In *6th Weimar Optimization and Stochastic Days 2009, October 15th-16th, Weimar, Germany*.

[9] Smola, A. J. and B. Schölkopf (2004). A tutorial on support vector regression. *Stat. Comput. 14*, 199–222.

[10] Vapnik, V. and O. Chapelle (2000). Bounds on error expectation for support vector machines. *Neural Comput. 12*(9), 2013–2036.

[11] Vapnik, V. N. (1995). *The Nature of Statistical Learning Theory*. Springer-Verlag, New York.