Surrogate modeling approximation using a mixture of experts based on EM joint estimation

Dimitri Bettebghor · Nathalie Bartoli · Stéphane Grihon · Joseph Morlier · Manuel Samuelides

Received: 9 February 2010 / Revised: 15 July 2010 / Accepted: 25 July 2010

Abstract An automatic method to combine several local surrogate models is presented. This method is intended to build accurate and smooth approximation of discontinuous functions that are to be used in structural optimization problems. It strongly relies on the Expectation−Maximization (EM) algorithm for Gaussian mixture models (GMM). To the end of regression, the inputs are clustered together with their output values by means of parameter estimation of the joint distribution. A local expert is then built (linear, quadratic, artificial neural network, moving least squares) on each cluster. Lastly, the local experts are combined using the Gaussian mixture model parameters found by the EM algorithm to obtain a global model. This method is tested over both mathematical test cases and an engineering optimization problem from aeronautics and is found to improve the accuracy of the approximation.

D. Bettebghor · S. Grihon
Airbus France (EDSAZBT), Toulouse, France

S. Grihon
e-mail: stephane.grihon@airbus.com

D. Bettebghor (✉) · N. Bartoli · M. Samuelides
ONERA (DTIM), Toulouse, France
e-mail: dimitri.bettebghor@onera.fr, dimitri.bettebghor@airbus.com

N. Bartoli
e-mail: nathalie.bartoli@onera.fr

J. Morlier · M. Samuelides
ISAE-SupAéro, Toulouse, France

J. Morlier
e-mail: joseph.morlier@isae.fr

M. Samuelides
e-mail: manuel.samuelides@isae.fr

Published online: 29 August 2010

Keywords EM clustering · Gaussian mixture models · Mixture of experts · Surrogate models

1 Introduction

In recent years, design engineers have been provided with many practical tools from mathematical optimization and most of the classical gradient-based optimization techniques are now widely used in all fields (see for instance Haftka and Gurdal 1992 in the field of structural optimization). When using these classical tools, one often faces long simulations times either to compute the optimization constraints, or the objective function, or both. To speed up design, approximation models were developed to tackle the slowness of repetitive code evaluations. When used within a design process, these approximation methods are often called surrogate models. Surrogate models arise from statistics and probability theory and are now widespread tools to approximate complicated functions. They are used inside the optimization process to approximate the objective function or the constraints, or they can directly approximate the results of the optimization process as a function of the optimization problem parameters (materials characteristics, load case in structural optimization for instance). They are also widely used in the multilevel and multidisciplinary optimization framework (Liu et al. 2004; Merval 2008; Merval et al. 2006). Surrogate modeling offers many ways to approximate functions from sample data: artificial neural networks (ANN) (Dreyfus 2005), moving least squares (MLS) (Nealen and Darmstadt 2004), radial basis functions (RBF) (Buhmann 2001), kriging (van Beers and Kleijnen 2004), support vector machines (Smola and Schölkopf 2004), multivariate adaptive regressive splines (MARS) (Friedman et al. 2001). A good

overview of the existing surrogate models can be found in Friedman et al. (2001), Kleijnen et al. (2005), Wang and Shan (2007) and Simpson et al. (2008). In Forrester and Keane (2009), Forrester and Keane provide an intensive review of most of the surrogate models and compare them when used within an optimization process. Nonetheless, one simple surrogate model might not be enough to approximate a complicated function, especially when this function features different behaviors depending on the region of the input space. This situation happens quite often in mechanics when computing critical buckling modes. For instance in Merval (2008), the optimization constraints to be approximated (reserve factors for skin buckling and local web stringer buckling for a composite stiffened panel) happen to be discontinuous and derivative-discontinuous which precludes the training of an accurate surrogate model. Indeed surrogate models usually assume that the function to approximate is smooth and are themselves smooth. This results in a high variance around the discontinuities and that makes the generalization power of the surrogate model poorer. One way to prevent this high variance would be to divide the input space into regions that do not feature discontinuities and then build a surrogate model on each of these regions. This way one could get rid of the discontinuities.

To improve the accuracy of a surrogate model, a common practice is to build several surrogate models on the same common learning basis. One is more likely to find a more accurate surrogate when building many surrogate models. As explained in Viana et al. (2009), several surrogate models prevents from building poorly fitted models. On the basis of the different surrogate models, one can choose the most accurate based on the classical statistical techniques of cross-validation and bootstrap to obtain estimates of the generalization error of the different surrogate models (see for instance Kohavi 1995 and Picard and Cook 1984). As pointed out in Acar and Rais-Rohani (2009), one of the drawbacks of choosing the best predictor is that we do not make use of all the resources used in the construction of the discarded surrogate models. The chosen predictor may be globally precise enough but may lack accuracy in some crucial areas of the input space (boundaries for instance), while one of several of the discarded surrogate models may perform better in these very areas. One could overcome this drawback by combining all the surrogate models by means of weights. This practice of combination relies on the same basis as committees of machines in artificial intelligence. A committee of machines is a collection of intelligent agents that vote and decide all together, hoping that errors would cancel as there are several experts. In the area of machine learning, this practice of combination appears in the bagging and boosting techniques. In the case of an ensemble of surrogate models, the different surrogate models may be simply averaged or weighted. Note that the weighting may be done globally (constant weights over the input space) as it is done in Viana et al. (2009) and in Acar and Rais-Rohani (2009) or locally (depending on the input space) as it is done in Zerpa et al. (2005) and in Sanchez et al. (2008). Even though the idea of combining several surrogate models seems appropriate to approximate functions, there is no evidence that combining is always better than selecting the best surrogate, as it is pointed out in Yang (2003).

Our approach is in the framework of ensembles of locally weighted surrogate models except that it is also based on a partitioning of the learning basis, whereas in most of the described techniques of ensembles, the surrogate models are built on the same common learning basis. Indeed, our concern is about the approximation of functions featuring discontinuities, heterogeneous behaviors and very different landscapes depending on the region of the input space. This is why, in our approach, a surrogate model is built over a specific region of the input space. From an optimization point of view, the global surrogate model needs to be continuous and even smooth (for gradient-based optimization techniques), this is why we combine them in a way that their errors are canceled, notably in the vicinity of discontinuities. As pointed out, this approach differs slightly from the existing ensembles of surrogate models, since each surrogate model, though applied over the whole input space at the very end, is only built over a specific region of the input space. Our approach is based both on the same idea as the committee of machines but also on the 'Divide and Conquer' principle. In the literature, this kind of approach is referred to as mixture of experts (MoE's). A general introduction to mixture of experts can be found in Friedman et al. (2001). One classical application is known as hierarchical mixture of experts (HME) and is described in Jordan and Jacobs (1994). In this study, Jordan and Jacobs present a general architecture of mixture of experts for supervised learning (regression and classification). This architecture is a tree structure where each nonterminal produces a soft split of the input value coming from the upper level. This soft split consists in giving different weights to the lower sub levels by means of a generalized linear model; the authors call it gating network. Each gating network produces soft splits until the terminal leaves, which produce output (real value for regression and binary for classification) by means of a generalized linear model. These terminal leaves are called by the authors expert network. The parameters of the different generalized linear models are estimated using a classical algorithm in statistics: the Expectation-Maximization algorithm (EM algorithm) on the input-output space, which means that in their study, partitioning and learning are based on the same algorithm. We propose a different method where clustering is separated from learning. The gating networks are not generalized linear models (GLIM) but Gaussian mixture models (GMM) still estimated through

EM algorithm. Based on the Gaussian mixture models estimates, the input-output space (or conjoint space) is clustered with respect to the maximum a posteriori (MAP). Once this clustering is done, we have a certain number of sub-bases and a surrogate model is trained over each sub-basis. This surrogate model can be quadratic regression, artificial neural networks and moving least squares regression, while Jordan and Jacobs use a generalized linear model. All the surrogate models are combined on the basis of the Gaussian parameters found by EM algorithm. The proposed technique relies on a certain number of hard tasks that try to answer the following questions

a) How do we cluster the learning basis? Clustering (or automatic classification) is one of most important areas of unsupervised learning. It aims at finding groups whose individuals are close in some sense into a collection of points. There are many different techniques to cluster data (K-means, K-medoids, quality threshold clustering, density clustering see Berkhin (2002) for instance where most of the classical algorithms for clustering are reviewed). Some of theses are hard clustering, where each point of the design space belongs to one and only one cluster. Others are fuzzy clustering, where each point of the design space belongs to several clusters and each point is associated to a random vector defining the probabilities to lie within each cluster.
b) Which local experts do we build and how do we combine them? As pointed out in Yang (2003), there is no reason that combining several surrogate will perform better than only one, which means that the combination has to be done carefully based on a technique that is expected to cancel errors.
c) How to choose the number K of clusters? This is a rather difficult question since there might not be perfect number of clusters. Indeed, we just want to find a good number of clusters such that each expert would do well enough to be combined. There might be different choices for K. The question of the number of clusters is central in clustering theory and involves different tools from statistics, probability theory and information theory. In Burnham and Anderson (2004), some of the most common criteria to find the best number of clusters (Akaike Information Criterion and the Bayesian Information Criterion) are thoroughly investigated and compared.

As said earlier, our concern is mostly about the approximation of discontinuous functions to the end of optimization. The clustering should be done such that the boundary between clusters would be as close as possible to the real discontinuities. We assume that these discontinuities may be distributed all over the domain and would naturally divide the input-output space into bunches of points that are connected but that may be anisotropic. Clustering should also be done in such a way that the we have a center and a parameterization of each cluster to combine the local surrogate models. To that end, we assume that a good representation of the conjoint data for discontinuous functions would be Gaussian mixture models. The EM algorithm would allow us to find good estimates of the Gaussian parameters. EM clustering for Gaussian mixture models gives an anisotropic representation of the data and therefore a parameterization of the clusters that makes it possible to combine the different surrogate models built on each cluster. In Bradley et al. (1998), some of the benefits that can be taken from using EM clustering are developed. Next, we describe our method by answering the former questions. We first present the theoretical background of our proposed technique: Gaussian mixtures models, clustering based on the GMM estimates in Section 2 and answer question a). We then focus on the combination of the local experts trained on the basis of the clustering, this will allow us to answer question b) and derive the original algorithm presented in Section 3 to improve the accuracy of surrogate modeling for discontinuous functions. In Section 4, we validate our proposed algorithm on test cases obtained from a discontinuous functions samples generator called Samgen and we also give a practical answer to question c). In Section 5, we finally test our proposed algorithm on a structural optimization problem from aeronautics. The EM algorithm, which is the basis of our method is recalled in Appendix A. We also give an original interpretation of a standard surrogate model the weighted least squares (WLS) in terms of mixture of experts based on a soft clustering and local quadratic experts in Appendix B. The tool called Samgen that we implemented to provide highly discontinuous functions in arbitrary dimension is outlined in Appendix C.

2 Gaussian mixture models and EM clustering

2.1 Gaussian mixture models

We first describe in this section Gaussian mixture models. Suppose we are given $\mathcal{X} = (x_1, ..., x_n)$ a set of data where $x_i \in \mathbb{R}^d$. Assume that these x_i's come from $\{X_i\}_{i=1...n}$ a set of identical and independently distributed (iid) random variables. An important problem in statistics and in probability theory is the estimation of the probability density function (pdf). In the case of GMM, we assume that the probability law of X is a weighted combination of a given number K of multivariate Gaussian laws

$$X \sim \sum_{k=1}^{K} \alpha_k \mathcal{N}(\mu_k, \Gamma_k), \tag{1}$$

where the α_k's are the mixture parameters, i.e, α_k is the proportion of the Gaussian k in the mixture. Note that, if we denote f_k the pdf of Gaussian k, the $\sim$ symbol means that the pdf of X is $\sum_{k=1}^{K} \alpha_k f_k$. We obviously have $\sum_{k=1}^{K} \alpha_k = 1$ since it is a probability density function. The other parameters are the Gaussian parameters. Recall that a multivariate Gaussian distribution is defined by its mean $\mu_k \in \mathbb{R}^k$ and its variance-covariance matrix $\Gamma_k \in \mathcal{M}_d(\mathbb{R})$, which is symmetric positive definite. The pdf of Gaussian $\mathcal{N}(\mu_k, \Gamma_k)$ is therefore

$$f_k(x) = \frac{1}{\sqrt{(2\pi)^d \det(\Gamma_k)}} e^{-\frac{1}{2}(x-\mu_k)^T \Gamma_k^{-1}(x-\mu_k)}. \tag{2}$$

To illustrate GMM's, we depicted Fig. 1 two different GMM's, respectively in $\mathbb{R}$ and in $\mathbb{R}^2$.

2.2 Expectation-maximization clustering

We recall in Appendix A the EM algorithm, which is a practical tool to determinate the underlying structure of data. More precisely in the Gaussian mixture model case, EM algorithm gives estimates of the means and the variance-covariance matrices. Since we consider regression problems, we assume we have $\mathcal{X} = (x_1, \ldots, x_n)$ a set of inputs and $\mathcal{Y} = (y_1, \ldots, y_n)$ the corresponding outputs. Note that we consider in the following that the output space is $\mathbb{R}$, but we could have considered $\mathbb{R}^d$ with no changes. We want to describe the conjoint law of (X, Y) as a GMM.

Consider the conjoint space $\mathcal{X} \times \mathcal{Y} = \mathcal{Z} = (z_1, \ldots, z_n)$ where $z_i = (x_i, y_i) \in \mathbb{R}^{d+1}$. Suppose we have set the number of clusters to K. We estimate through EM algorithm the parameters of the K multivariate Gaussian distributions in $\mathbb{R}^{d+1}$ such that

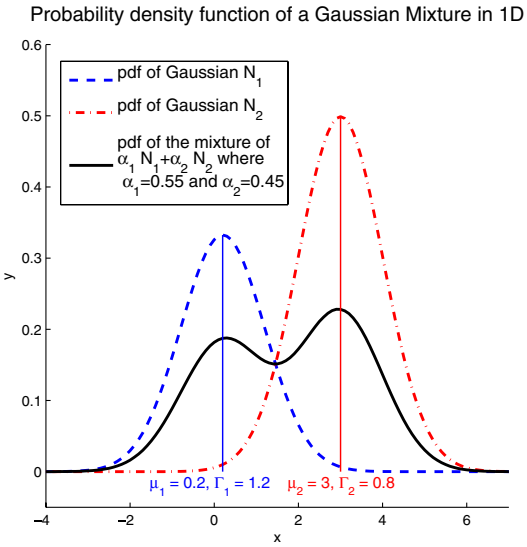
$$Z \sim \sum_{k=1}^{K} \alpha_k \mathcal{N}(\mu_k, \Gamma_k), \tag{3}$$

where the α_k's are the mixture parameters, i.e for all $k \in i \ldots K$, $\alpha_k \in [0, 1]$ and

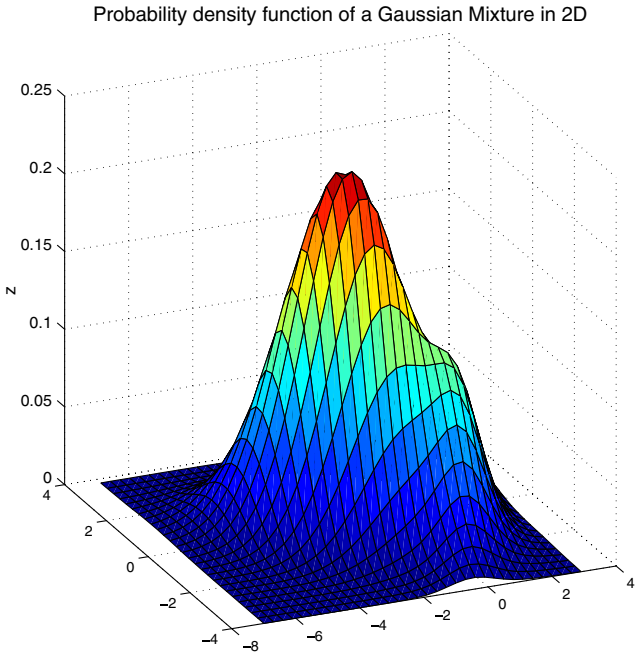
$$\sum_{k=1}^{K} \alpha_k = 1 \tag{4}$$

and $\mu_k \in \mathbb{R}^{d+1}$ is the mean of the Gaussian distribution k and denote

$$\mu_k = \begin{pmatrix} \mu_k^X \\ \mu_k^Y \end{pmatrix}, \tag{5}$$



(a) pdf of a GMM in $\mathbb{R}$



(b) pdf of a GMM in $\mathbb{R}^2$

Fig. 1 Examples of GMM pdf's

where $\mu_k^X \in \mathbb{R}^d$ is the $X-$coordinates of the mean μ_k and $\mu_k^Y \in \mathbb{R}$ is the $Y-$coordinate of the mean μ_k. $\Gamma_k \in \mathcal{M}_{d+1}(\mathbb{R})$ is the variance-covariance matrix and denote

$$\Gamma_k = \begin{pmatrix} \Gamma_k^X & \nu_k \\ \nu_k^T & \xi_k \end{pmatrix}, \tag{6}$$

where $\Gamma_k^X \in \mathcal{M}_d(\mathbb{R})$ is the variance-covariance matrix of X for Gaussian k, $\nu_k \in \mathbb{R}^d$ is $\mathrm{Cov}(X, Y)$ for Gaussian k and $\xi_k = \mathrm{Var}(Y)$.

Once the GMM parameters are estimated, we can now compute the posterior probabilities, that is to say, the probability for a given $(x, y) \in \mathbb{R}^{d+1}$ to lie within cluster k_i. It is given by Bayes' formula where κ denotes the discrete random variable associated with the clusters

$$\mathbb{P}(\kappa = k_i | (X, Y) = (x, y)) = \frac{\mathbb{P}(\kappa = k_i)\mathbb{P}((X, Y) = (x, y) | \kappa = k_i)}{\sum_{k=1}^{K} \mathbb{P}(\kappa = k)\mathbb{P}((X, Y) = (x, y) | \kappa = k)}.$$

For the particular case where (X, Y) is assumed to be a Gaussian mixture model: $(X, Y) \sim \sum_{k=1}^{K} \alpha_k \mathcal{N}(\mu_k, \Gamma_k)$ we have for all $k \in \{1, \ldots, K\}$

$$\mathbb{P}(\kappa = k) = \alpha_k,$$
$$(X, Y) | \kappa = k \sim \mathcal{N}(\mu_k, \Gamma_k),$$

which leads with $z = (x, y)$ to

$$\mathbb{P}(\kappa = k_i | (X, Y) = (x, y)) = \frac{\det(\Gamma_{k_i})^{-\frac{1}{2}} \alpha_{k_i} e^{-\frac{1}{2}(z-\mu_{k_i})^T \Gamma_{k_i}^{-1}(z-\mu_{k_i})}}{\sum_{k=1}^{K} \det(\Gamma_k)^{-\frac{1}{2}} \alpha_k e^{-\frac{1}{2}(z-\mu_k)^T \Gamma_k^{-1}(z-\mu_k)}}. \quad (7)$$

Note that in (7), the 2π factor vanishes, but the determinant of the variance-covariance matrices Γ_k's remain as the mixture parameters α_k's. Formula (7) offers two ways of partitioning the data set:

- hard clustering: we simply choose over all the k's the one that gives the highest probability. Given (x, y) in the conjoint space, (x, y) lies in cluster j where

$$j = \text{argmax}_{j=1,\ldots,K} \mathbb{P}(\kappa = j | (X, Y) = (x, y)). \quad (8)$$

- soft clustering: points belong to all clusters, we assign to each point (x_i, y_i) its probability to be generated knowing the cluster j (and thus the mode j). Each point is therefore given a set of K probabilities $p = (p_1, \ldots, p_K)$ where

$$p_j = \frac{\mathbb{P}((X, Y) = (x, y) | \kappa = j)}{\sum_{k=1}^{K} \mathbb{P}((X, Y) = (x, y) | \kappa = k)}. \quad (9)$$

  Indeed, we want to train a surrogate model over $\mathcal{X}$ for mode j, therefore each learning point x_i should be weighted by the probability p_j to have x_i knowing that the cluster is j. This probability is normalized such that $\sum_{k=1}^{K} p_k = 1$.

In this article, we chose the hard clustering. The hard clustering keeps the localization of the examples in the same cluster in the sense that they are close with respect to the Mahalanobis distance[1] associated with the variance-covariance matrix Γ_k. The other reason is that building surrogate model from weighted examples is not straightforward, apart from linear and polynomial multivariate regressions where it boils down to weighted least squares (see Appendix B). In the next section, we describe the way we combine the different local experts built on this hard clustering.

3 Combining local experts and proposed algorithm

The learning basis is split into K learning sub-bases and an expert f_k is trained over each sub-basis k. Any surrogate model can be used, we give results obtained using the following different local experts:

- linear regression: the most simple expert (apart from the constant expert which leads to radial basis functions regression). It can be computed easily and the multivariate linear regression parameters are directly given by the Gaussian component parameters

$$f_k(x) = \frac{\text{Cov}^k(X, Y)}{\text{Var}^k(X)}\left(x - \mathbb{E}^k(X)\right) + \mathbb{E}^k(Y) = (\Gamma_k^X)^{-1} \nu_k \left(x - \mu_k^X\right) + \mu_k^Y. \quad (10)$$

  In that case, once the EM algorithm is done, all the parameters of the MoE are computed. In this particular case, clustering and learning are not separated as in Jordan and Jacobs (1994). This MoE is therefore very cheap to compute. Note that a numerical instability (apart from EM algorithm that might converge very slowly) can arise from the inversion of the variance-covariance matrices Γ_k (for clustering) and Γ_k^X (to build local experts and combine them). This should be done carefully using for instance QR factorization.[2]
- quadratic regression: the original response surfaces, which are quadratic polynomials over $\mathbb{R}^d$ as extensively described in Myers et al. (2009). They are also relatively inexpensive to build however there is no easy

[1] The Mahalanobis distance of a random variable $X \in \mathbb{R}^d$ is the distance defined by the inverse the variance-covariance matrix $\Gamma = \text{Var}(X)$: for $\omega_1, \omega_2 \in \mathbb{R}^d$ the Mahalanobis distance is $D_M(\omega_1, \omega_2) = ||\omega_1 - \omega_2||_{\Gamma^{-1}} = \sqrt{(\omega_1 - \omega_2)^T \Gamma^{-1} (\omega_1 - \omega_2)}$. It does define a distance since the inverse of Γ (sometimes called the precision matrix) is symmetric positive definite.

[2] All these matrices are symmetric positive definite but they can become nearly-singular especially in case of redundant data (linearity), QR factorization performs better than Gaussian reduction and even Choleski factorization.

formula that can be derived from the Gaussian components parameters. In our case, we computed it in a simple way taking care of the inversion of the system.

- artificial neural networks: we use here the classical Multi Layer Perceptron (MLP) as a local expert; MLP models are thoroughly described in Haykin (2008). We use one-layer networks and the number of hidden neurons is classically determined through a cross-validation procedure and the network is trained using Levenberg−Marquardt algorithm.
- moving least squares: the MLS expert is the most complicated expert to compute for it is an implicit model that needs to be recomputed at each new evaluation point. We implemented a moving least squares method based on the Backus−Gilbert approach that can be found in Fasshauer (2005). We also implemented a golden ratio search to optimize the hyper-parameter σ (width of the Gaussian kernel, see Appendix B). A landmark paper on MLS is Levin (1998) and a brief introduction can be found in Nealen and Darmstadt (2004).

Nonetheless, any surrogate model can be used (kriging, support vector regression, radial basis function, multivariate adapted regressive splines) as a local expert and can be perfectly improved using an ensemble of surrogate models, or boosting (Meir and Ratsch 2003). Moreover, a local expert can be a black-box or itself an MoE and so on.

Once we build our local experts f_k, we want to predict the response y for a new entry $x \in \mathbb{R}^d$. This is done by combining them. We form a linear combination of the local experts f_k. A natural idea is that this linear combination should not be constant over the whole input space and should give more weight to expert f_k when x gets closer to the center of cluster k with respect to the natural Mahalanobis distance inherited by the variance-covariance matrix of Gaussian k. Namely the global model $\hat{f}$ is

$$\hat{f}(x) = \sum_{i=1}^{K} \beta_k(x) f_k(x), \tag{11}$$

where $\beta = (\beta_1, \ldots, \beta_K)$ is an expert that gives the local weights (local in the sense that it does depend on x). A natural gating network would be

$$\beta(x) = (\mathbb{P}(\kappa = 1|X = x), \ldots, \mathbb{P}(\kappa = K|X = x)), \tag{12}$$

such that the global model would be

$$\hat{f}(x) = \sum_{i=1}^{K} \mathbb{P}(\kappa = i|X = x) f_i(x). \tag{13}$$

Equation (13) is the classical probability expression of mixture of experts (as it can be found in Jordan and Jacobs 1994). Note that this expression may represent a lot of different situations and therefore a lot of different MoE's. For instance, as said earlier, the weighted least squares can be interpreted as an Moe using that equation (see Appendix B). To use (13) we need to know what is the law of κ knowing that $X = x$ and without knowing that $Y = y$, this can be easily obtained with the Gaussian parameters found by EM algorithm. Indeed, from the conjoint law $(X, Y) \sim \sum_{k=1}^{K} \alpha_k \mathcal{N}(\mu_k, \Gamma_k)$, we can derive the law of $X|\kappa = k$ without knowing Y

$$X|\kappa = k \sim \mathcal{N}_d(\mu_k^X, \Gamma_k^X), \tag{14}$$

such that the global GMM law of X is

$$X \sim \sum_{k=1}^{K} \alpha_k \mathcal{N}(\mu_k^X, \Gamma_k^X). \tag{15}$$

Note that this Gaussian mixture model is different from the one we would have obtained by applying EM only on the inputs X's for it is the projection on the input space of the conjoint law. Therefore we can derive equivalently the posterior probability from Bayes' formula

$$\mathbb{P}(\kappa = k_i|X = x) = \frac{\det\left(\Gamma_{k_i}^X\right)^{-\frac{1}{2}} \alpha_{k_i} e^{-\frac{1}{2}(x-\mu_{k_i}^X)^T \Gamma_{k_i}^{X-1}(x-\mu_{k_i}^X)}}{\sum_{k=1}^{K} \det(\Gamma_k^X)^{-\frac{1}{2}} \alpha_k e^{-\frac{1}{2}(x-\mu_k^X)^T \Gamma_k^{X-1}(x-\mu_k^X)}}. \tag{16}$$

Note that the global model defined with (13) and (16) is completely smooth. In the sequel, this mixture of experts will be referred to as smooth mixture. At this point, we can think of another way of combining the local surrogate models that takes more avantage from the clustering made by the EM algorithm. Indeed, based on the Gaussian parameters estimated on the clustering step, we can predict which cluster a new given entry x lies in and simply assign to this entry x the corresponding local surrogate model. This means that we can, at least formally, define a partitionning of the whole input space: $\mathcal{X} = \cup_{k=1}^{K} \mathcal{X}_i$ and simply define the law of k knowing that $X = x$ as a uniform discrete law such that the global model would be

$$\hat{f}(x) = \sum_{k=1}^{K} \mathbb{1}_{\mathrm{Cl}_k^X}(x) f_k(x) \tag{17}$$

where

$$\mathbb{1}_{\mathrm{Cl}_k^X}(x) = \begin{cases} 1 & \text{if} \quad k = \operatorname{argmax}_{j=1,\ldots,K} \mathbb{P}(\kappa = j|X = x) \\ 0 & \text{if} \quad k \neq \operatorname{argmax}_{j=1,\ldots,K} \mathbb{P}(\kappa = j|X = x) \end{cases} \tag{18}$$

and (17) defines the most simple mixture of experts where a new entry x is given a cluster k and the predicted value

is simply $f_k(x)$. An important feature of this mixture of experts is that it is not continuous, indeed at the boundary between two adjacent clusters[3] the two adjacent local surrogate models need not to match resulting in a globally discontinuous model. This mixture of experts will be referred to as hard mixture of experts. In the case when the functions to approximate is discontinuous the hard mixture version is likely to be more accurate than the smooth version. Besides being discontinuous, the hard mixture version may create artificial discontinuities where the original function does not have ones (see Fig. 2d). In this article, we are mainly concerned with approximating functions that are objective or constraints functions of an optimization problem that is to be solved on the basis of a gradient-based method, this is why we will focus on the smooth mixture of experts. Indeed, in such applications, we are not only concerned with the accuracy of the approximation but also with the regularization of the original approximation. In terms of accuracy the hard version is likely to perform better (see for instance Fig. 2d) but the optimization algorithm may fail to converge due to the non-differentiability of the global approximation model.

Our algorithm is presented here and results of this method are given in the following sections with the help of two test cases and one engineering problem. This method is also illustrated on a one-dimensional test case in Fig. 2.

1. Assemble Z the conjoint learning basis where $z_i \in \mathbb{R}^{d+1}$ contains inputs $x_i \in \mathbb{R}^d$ and output $y_i \in \mathbb{R}$, see Fig. 2a

$$Z = \begin{pmatrix} x_1^{(1)} & \dots & x_n^{(1)} \\ \vdots & & \vdots \\ x_1^{(d)} & \dots & x_n^{(d)} \\ y_1 & \dots & y_n \end{pmatrix}. \tag{19}$$

2. Set the number of clusters K as explained below. In Fig. 2 the number of clusters was set to 3.
3. Apply EM algorithm to Z with K to get $\hat{\alpha}_k$, $\hat{\mu}_k$ and $\hat{\Gamma}_k$, estimates of the real Gaussian parameters.
4. Hard clustering of the data, see Fig. 2b. $z_i = (x_i, y_i)$ belongs to cluster k_i where

$$k_i = \text{argmax}_{j=1,\dots,K} \mathbb{P}(j|(X,Y) = (x_i, y_i)). \tag{20}$$

   where $\mathbb{P}(\kappa = j|(X,Y) = (x_i, y_i))$ is computed using (7)
5. Split the conjoint learning basis into K clusters $Z = \cup_{i=1}^K Z^{(i)}$.
6. For $i = 1 \dots K$
   (a) Remove outliers using for instance Mahalanobis distance.
   (b) Split randomly $Z^{(i)} = Z^{(i)}_{\text{learn}} \cup Z^{(i)}_{\text{test}}$ into learning basis and test basis.
   (c) Train expert f_i on $Z^{(i)}_{\text{learn}}$, choose the best expert f_i with $Z^{(i)}_{\text{test}}$, see Fig. 2c.
7. Combine all the experts with

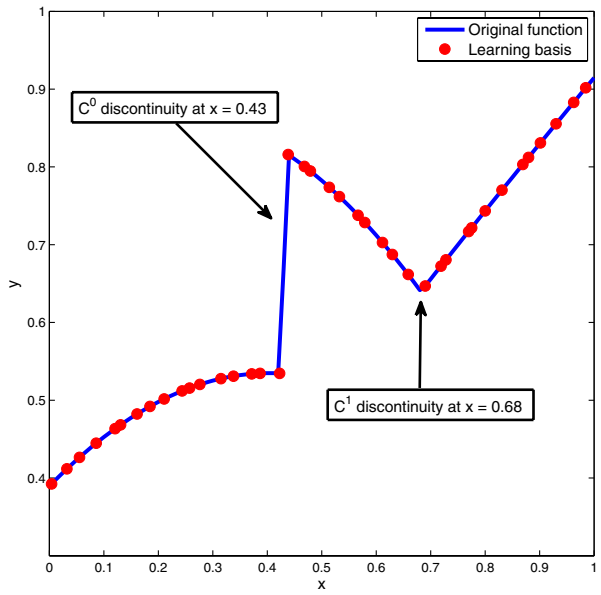
$$\hat{f}(x) = \sum_{i=1}^{K} \mathbb{P}(\kappa = i|X = x) f_i(x). \tag{21}$$

   where $\mathbb{P}(k = i|X = x)$ is computed using (16), see Fig. 2d where we also plotted the hard version of the mixture of experts.

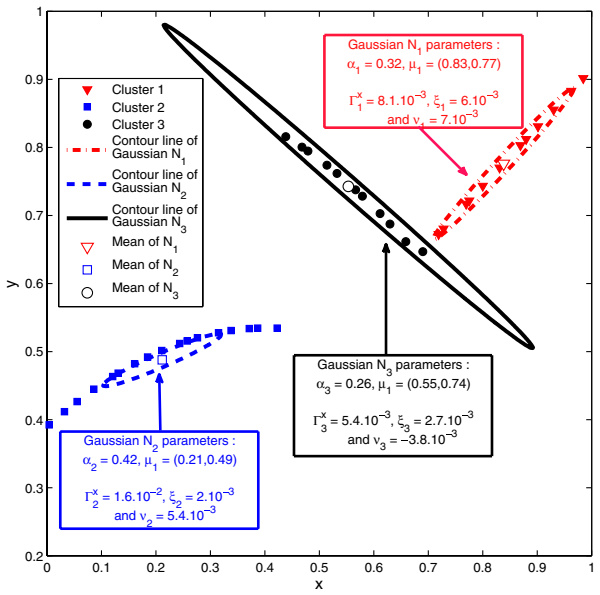
So far we have not answered the crucial question about the number of clusters. We suggest here a practical way to determinate a good number of such clusters. As discussed earlier, there might not be a perfect number of clusters, the function may be continuous and non derivative discontinuous and only one expert may be a good choice to approximate the function. Nonetheless, we usually observed even in that case that several experts can perform better than only one. On the other hand, too many experts may split the learning basis onto clusters with too few examples to train a reasonable surrogate model. There is a trade-off between the complexity of the model (number of free parameters) and the capacity of generalization of the MoE. To set the correct number of clusters (and experts) we suggest to build MoE's with linear experts and quadratic experts, estimate the MSE and the other error measures for different number of clusters, (say for $k = 2 \dots N_A/10$) and then choose the number of clusters that minimizes the MSE and $\hat{E}_{\text{mean}}$. In the case where there are several number of clusters that could do it, we would better choose the lowest to make the clusters as big as possible and thus improve the accuracy of the local experts. Part of the reason for this procedure is that building linear experts and quadratic experts is inexpensive in comparison to building neural experts or moving least squares regression experts.

Note that this algorithm was designed for smoothing discontinuous or derivative-discontinuous functions. It was pointed out that the EM clustering is expected to separate disconnected parts of the conjoint space and therefore to grasp regions where the response to approximate is continuous and where building a surrogate model would be much easier than building one on the whole domain. Nevertheless, combining all the local surrogate models into a global smooth one is likely to make the approximation less accurate around the discontinuities, while in the hard mixture version the discontinuities are preserved. This is

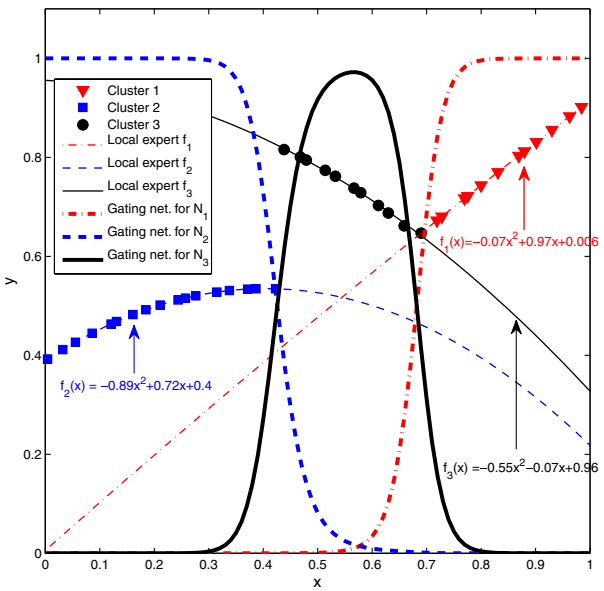
[3] This boundary is often known in Probability as the Bayes classifier.



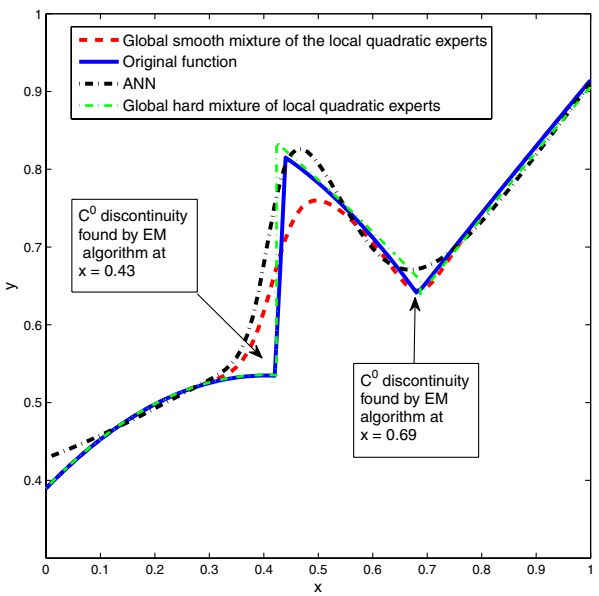
(a) samgen_1D_1 test case and learning basis.



(b) Results of the EM clustering and associated Gaussian laws



(c) Local quadratic experts and gating networks



(d) Global predictor

Fig. 2 Sketch of the proposed algorithm for a 1D test case `samgen_1D_1`. **a** Original function and learning basis. Note that `samgen_1D_1` features two $\mathcal{C}^0$ and $\mathcal{C}^1$ discontinuities. EM clustering is expected to separate them well enough to locally build an accurate surrogate model. **b** EM clustering on the learning basis for $K = 3$. We depict the *contour lines* of the quadratic function associated to the variance-covariance matrix Γ_i at 3×10^{-6} for $i = 1 \ldots 3$. Note that these lines are simply the balls of radius 3×10^{-6} centered at μ_i for the Mahalanobis distance associated with the variance-covariance matrices $\begin{pmatrix} \Gamma_i^X & \nu_i \\ \nu_i & \xi_i \end{pmatrix}$. **c** Local quadratic experts and gating networks. See that the gating network associated to cluster 2 is quite steep. This is due to the relative small size of cluster 2. **d** Global surrogate models obtained through the mixture of local quadratic experts. We plotted the soft and the hard mixture versions. We also depicted a reasonably good artificial neural network to compare. We observe that the smooth predictor is very accurate on clusters 1 and 3 is a little bit less accurate on cluster 2. While the artificial neural network does not generalize very well at the boundaries of the domain, the global smooth predictor performs better at these boundaries due to the local behaviour of the surrogate models. The hard mixture predictor is much more accurate since it does not regularize the discontinuities. It creates though an artificial $\mathcal{C}^0$ discontinuity at $x = 0.69$

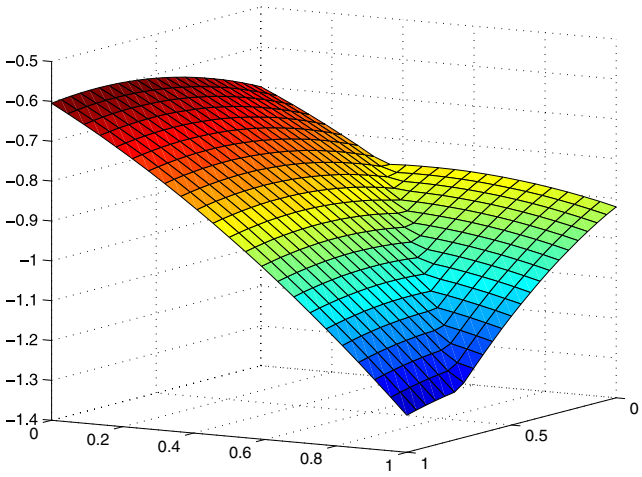
illustrated in Fig. 2d. Besides, such functions may be well approximated using several existing methods. Apart from the methods of ensemble of surrogate model that we briefly describe in the introduction section, the multivariate adaptive regressive splines (MARS) are often said to handle such discontinuous functions especially in high dimensions (see Friedman et al. 2001). Indeed, MARS builds a model using a basis of piecewise linear functions and their products. Nevertheless, MARS does not cluster the learning basis and does not seem to keep the local behavior of the function to be approximated. One of the benefits of our proposed algorithm is to subdivide the whole design space into 'regions of interest' where previous knowledge or expertise for a specific surrogate model may help to build an accurate local approximation that will be merged into a global approximation. Indeed, we can use cross-validation when building each local expert and therefore get estimators of the generalization error made by each expert. This information can be used to detect 'regions' (that can be a sole cluster or several) where this error is larger than in the rest of the input space. These regions may be enriched (by adding learning points) to get a better global approximation model. Lastly, this local information can be compared to the generalization error of the global approximation model to assess the accuracy of the mixture of experts.

4 Validation on test cases

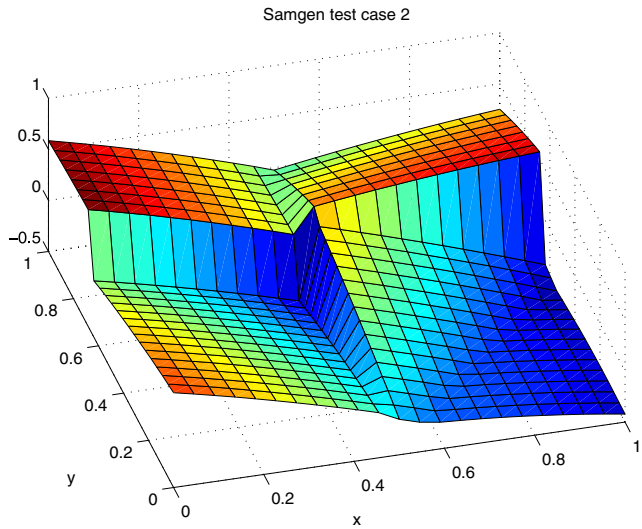
We present the results of our proposed algorithm obtained with three cases. The two first are mathematical examples who were designed to feature $\mathcal{C}^0$ and $\mathcal{C}^1$ discontinuities. These two functions were generated with Samgen (see Appendix C). Part of this study was to carry out new methods to approximate discontinuous functions or functions that feature different global behaviors depending on the region of the design space. These two examples `samgen_2D_1` and `samgen_2D_2` fulfill these requirements.

These two functions are defined through the maximum of non-convex quadratic functions over sub-domains of $[0, 1]^2$. We depicted these two functions Fig. 3. Note that `samgen_2D_1` only features a discontinuity of the derivative, `samgen_2D_2` features two orthogonal discontinuities that subdivide the whole domain into four sub-domains, and it also features a discontinuity of the derivative. These test cases are interesting for several reasons.

- Most surrogate models tend to regularize the discontinuities and therefore hardly handles this test case.
- The outputs are in the same range for all sub-domains. This is why clustering only on the output values does not detect the correct four sub-domains. We expect EM clustering to detect these four sub-domains.



(a) samgen_2D_1 test case



(b) samgen_2D_2 test case

Fig. 3 Mathematical test cases generated with Samgen, outlined in Appendix C. These functions features $\mathcal{C}^0$ discontinuities (**b**) and $\mathcal{C}^1$ discontinuities (**a** and **b**)

To test our proposed algorithm, we used the same procedure for both test cases. The general idea was to compare the 'best' surrogate model over the whole domain and the proposed MoE for different types of local experts.

To assess the sensitivity of the clustering to design of experiments, we generated several designs of experiments, ran on each design the proposed algorithm and compared the MoE obtained with the 'best' surrogate over the whole domain. At the end, we compared the mean of the different

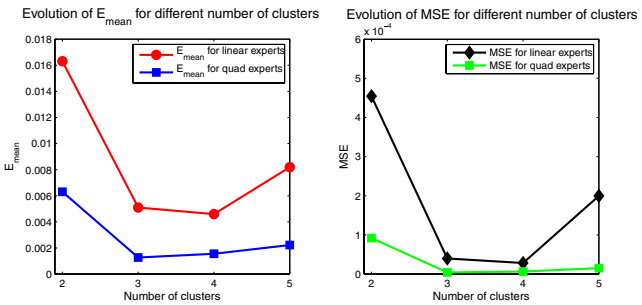
error measures. More formally, our procedure was the following

1. We first set the number of learning points N_A.
2. We generated N_{doe} designs of experiments.
3. For each design of experiment, we built a reasonably good surrogate model over the whole domain found after a thorough investigation. To that end, we trained ANN, MLS, quadratic and linear models and chose the one that gives the least generalization error estimated through a classical cross-validation procedure.
4. For each design of experiment, we ran the proposed algorithm. We decided to compare five different types of MoE's
   - MOE_LIN: the local experts are linear regression experts
   - MOE_QUAD: the local experts are quadratic regression experts
   - MOE_ANN: the local experts are artificial neural Networks
   - MOE_MLS: the local experts are moving least squares experts.
   - MOE_BEST: once the clustering is done, all the different types of experts are build and the best expert over each cluster is chosen, resulting in a global mixture of experts made of different types of experts.
5. We compared of all the surrogate models built using the following measures of approximation quality
   - RMSE: root mean squared error, most common measure (the surrogate models are usually trained by minimizing this quantity). Note that we could have used the mean squared error (MSE), since, in this very case, we just use the RMSE to compare different surrogate models among them. Owing to the monotonicity of the square root function, both RMSE and MSE would give the same rank among them.

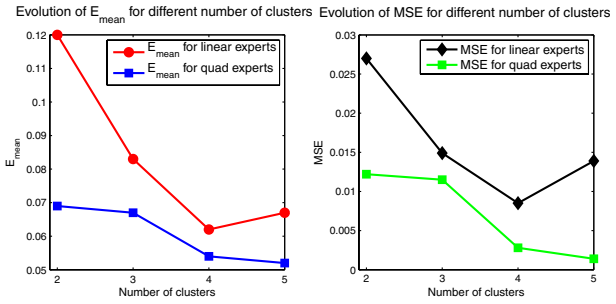
$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} ||f(x_i) - y_i||^2} \tag{22}$$

   - $\hat{E}_{\max}$: maximum of the absolute error
   - $\hat{E}_{\text{mean}}$: mean of the absolute error. We did not use the relative error measure to prevent from dividing by zero since `samgen_2D_2` maps to $[-1, 1]$
6. Finally, we computed the mean for each error measure to compare all the different surrogate models.

For `samgen_2D_1`, $N_{\text{doe}} = 100$ Latin hyper-cube sampling (LHS) designs of experiments of $N_A = 80$ points in $[0, 1]^2$ were generated. The global surrogate models and the proposed MoE's local experts were trained over a classical cross-validation partition of $(4/5, 1/5)$ for learning basis and test basis. The errors presented here were computed over a full factorial plan of 35×35 points and the means of the errors are computed over the 100 LHS. The number of clusters was set to 3 based on the practical procedure we describe at the end of Section 3. We plotted the MSE and $\hat{E}_{\text{mean}}$ for linear and quadratic expert for different number of clusters in Fig. 4a. We can see that in the case of `samgen_2D_1` that three or four clusters is a good choice. Therefore, we chose three clusters in that case. Results for the different MoE's are given in Table 1. For more than 85 of the different LHS designs, one or several MoE's happened to perform better than the best surrogate model over the whole domain, which happened to be an ANN in most cases. In average, the 'best' surrogate model over the whole domain performs pretty much as well as MOE_QUAD. In this test case, we see that the best surrogate model is MOE_ANN. Note that MOE_MLS is slightly less accurate. Note also that the MOE_BEST had in most cases ANN and MLS experts as best experts and that it performs as well as MOE_ANN. The reasons why the best experts are not quadratic (while the true function is defined as a maximum of quadratic function) is that the EM clustering is not perfect with respect to the true nature of the



(a) samgen_2D_1 test case



(b) samgen_2D_2 test case

Fig. 4 $\hat{E}_{\text{mean}}$ and MSE for different number of clusters: **a** `samgen_2D_1` test case, **b** `samgen_2D_2` test cases

Table 1 Results for `samgen_2D_1`

Type	RMSE (%)	$\hat{E}_{\max}$	$\hat{E}_{\text{mean}}$
Best surrogate model over the whole domain			
ANN	0.59	1.2×10^{-2}	7.1×10^{-3}
Proposed MoE			
MOE_LIN	2.66	1.7×10^{-2}	6.6×10^{-3}
MOE_QUAD	0.52	1.1×10^{-2}	2.8×10^{-3}
MOE_ANN	0.73	6×10^{-3}	1.8×10^{-3}
MOE_MLS	0.31	9.7×10^{-3}	3.2×10^{-3}
MOE_BEST	0.29	7.3×10^{-3}	2.9×10^{-3}

Table 2 Results for `samgen_2D_2`

Type	RMSE (%)	$\hat{E}_{\max}$	$\hat{E}_{\text{mean}}$
Best surrogate model over the whole domain			
ANN	13	2.14	0.54
Proposed MoE			
MOE_LIN	12.6	0.51	0.5
MOE_QUAD	9.54	0.36	4.5×10^{-2}
MOE_ANN	6.86	0.23	3.7×10^{-2}
MOE_MLS	5.66	0.26	2.4×10^{-2}
MOE_BEST	5.29	0.29	3.1×10^{-2}

function. A few learning points may be misclassified and ANN are more likely to perform better with outliers than quadratic models.

For `samgen_2D_2` $N_{\text{doe}} = 100$ LHS designs of experiments of $N_A = 100$ points in $[0, 1]^2$ were generated. More learning points were used to get rid of poorly fitted models that were due to the important discontinuities. The global surrogate models and the proposed MoE's local experts were trained over a classical cross-validation partition of $(4/5, 1/5)$ for learning basis and test basis. The errors presented here were computed over a full factorial plan of 50×50 points (to be sure that we also consider the error near the discontinuities) and the means of the error measures were computed over the 100 plans. The number of clusters was set to 4 on the basis of the procedure described at the end of Section 3. We plotted the MSE and $\hat{E}_{\text{mean}}$ for linear and quadratic expert for different number of clusters in Fig. 4b. We can see that in the case of `samgen_2D_2` that four or five clusters is a good choice. Therefore, we chose four clusters in that case. We also depicted in Fig. 5 the gating networks α_i's we obtained in that case. Results for the different MoE's are drawn in Table 2. We observed that all of the different MoE's performed better than the best surrogate model (which was also an ANN for 90 of the 100 designs, the rest being MLS). Results show that a simple and very inexpensive MOE_LIN performs better than a sophisticated ANN. Regarding the different types MoE's, we clearly see that the best is again MOE_ANN with MOE_MLS being slightly less accurate and MOE_BEST is again very close to MOE_ANN.

In both cases, MOE_BEST does not give the best accuracy, it is however very close to the best MoE. This may

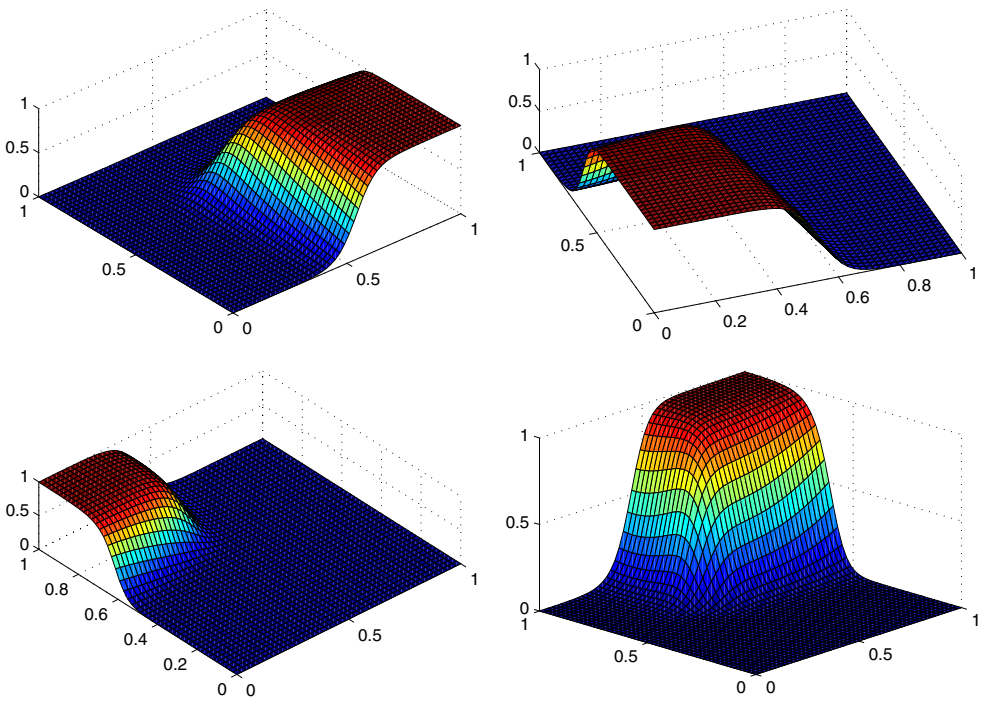


Fig. 5 Gating networks for $K = 4$ clusters in the case of `samgen_2D_2`. They are very close to the real C^0 discontinuities

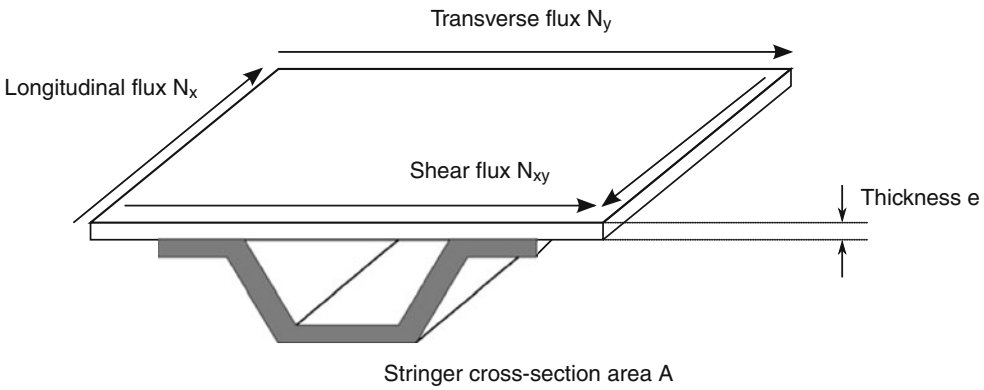


Fig. 6 $\Omega-$shaped composite super-stringer under the load case (N_x, N_y, N_{xy})

be due to the fact a very good local expert may not generalize very well outside the cluster where it was built. The best expert is somewhat too local and in these test cases, it seems like combining different types of local experts in the same global MoE does not improve the accuracy. This may also be due to the definition of the test cases, which feature the same kind of quadratic function over each domain (see Appendix C on Samgen). For other test cases, where the function behaves differently depending on the region (polynomial on one region and non-polynomial on another one for instance), MOE_BEST would certainly perform better. Is is worth noting that we also observed that for a given type of MoE, the variance of the error measure over the 100 learning bases was quite low, which would indicate that once the type of the expert has been chosen, the accuracy will be nearly independent of the clustering. It should be indicated that in some cases where EM algorithm could not find good estimates and failed to converge (or converged too slowly). Due to the fact there were insufficient example some clusters were almost degenerated which precluded the training of a correct local expert (apart from the linear and quadratic regressions). We must therefore insist in the fact that this procedure needs a certain number of examples and could not be applied in cases where there are very few examples.

5 Stiffened panel optimization test case

We turn now to the third case, which is an engineering problem of structural optimization from aeronautics. It consists of the sizing of a composite panel reinforced by an Ω-shaped stringer, as depicted in Fig. 6. More precisely, we are given N_x, N_y, N_{xy}, the longitudinal flux, the transverse flux and the shear flux. Under this load case, we want to minimize the cross section area of the stringer A and the thickness of the panel e under hard structural constraints (local buckling of the web of the stringer, Euler buckling, skin buckling). These constraints are computed with in-house tools based both on analytical formulas and a Rayleigh$-$Ritz approach. We can try to approximate the skill tool, but we are interested here in approximating directly the results of the optimization.[4] Given N_x, N_y, N_{xy} we want to estimate the optimums A^* and e^*.

To that purpose, the input space, was naturally divided into four sub-domains and 4,000 examples were computed with Boss Quattro (LHS of 1,000 points for each sub-domains)

- $D_1 = \{(N_x, N_y, N_{xy}) | N_x > 0, N_y > 0\}$
- $D_2 = \{(N_x, N_y, N_{xy}) | N_x > 0, N_y < 0\}$
- $D_3 = \{(N_x, N_y, N_{xy}) | N_x < 0, N_y > 0\}$
- $D_4 = \{(N_x, N_y, N_{xy}) | N_x < 0, N_y < 0\}$

This partition of the input space makes sense from physical considerations. Indeed, the shear flux N_{xy} does not influence that much the optimization and we generally observe in mechanics different behaviors depending whether the flux is positive (tension) or negative (compression). We want to find a good approximation of the optimization over each sub-domain. Note that this physical subdivision is in itself a clustering and the resulting global model which consists in assembling all the local experts is a MoE as described in (17). We apply the proposed technique over each sub-domain (and the final global model will be a mixture of mixture of experts) and compare the results with the best surrogate over the whole sub-domain.

Results are depicted in Tables 3, 4, 5, 6 for sub-domains and the overall results are presented in Table 7. Note that we only included results for MOE_ANN since this MoE usually performs slightly better than the other MoE's. We also changed our error measures; we considered the relative error instead of the absolute error and also include the $\alpha-$quantile for $\alpha = 5$ and $\alpha = 1$, e.g the $\alpha-$quantile is defined as the percentage of test points that are below an error of $\alpha\%$. By using the relative error, the output values

[4] Such regressions are mainly used to speed up pre-sizing of the aircraft and are known as design curves.

Table 3 Results for D_1

Output	Type	$\hat{E}_{\max}$ (%)	$\hat{E}_{\text{mean}}$ (%)	$\alpha = 1\%$ (%)	$\alpha = 5\%$ (%)
Best surrogate model over the whole domain D_1					
A^*	ANN	11	1.2	69	96
e^*	MLS	14.7	1.3	63	95
Proposed MoE with ANN experts ($K = 4$)					
A^*	MOE_ANN	5.6	0.8	80	98
e^*	MOE_ANN	9.5	1.04	65	98

Table 4 Results for D_2

Output	Type	$\hat{E}_{\max}$ (%)	$\hat{E}_{\text{mean}}$ (%)	$\alpha = 1\%$ (%)	$\alpha = 5\%$ (%)
Best surrogate model over the whole domain D_2					
A^*	MLS	36.5	2.3	52	88
e^*	MLS	25.7	2.6	41	84
Proposed MoE with ANN experts ($K = 5$)					
A^*	MOE_ANN	17.5	1.3	71	95
e^*	MOE_ANN	19.5	1.8	63	89

Table 5 Results for D_3

Output	Type	$\hat{E}_{\max}$ (%)	$\hat{E}_{\text{mean}}$ (%)	$\alpha = 1\%$ (%)	$\alpha = 5\%$ (%)
Best surrogate model over the whole domain D_3					
A^*	MLS	47	3.4	34	83
e^*	ANN	30.5	2.9	38	84
MoE with ANN experts ($K = 6$)					
A^*	MOE_ANN	20.5	2.8	50	81
e^*	MOE_ANN	15.1	1.9	44	89

Table 6 Results for D_4

Output	Type	$\hat{E}_{\max}$ (%)	$\hat{E}_{\text{mean}}$ (%)	$\alpha = 1\%$ (%)	$\alpha = 5\%$ (%)
Best surrogate model over the whole domain D_4					
A^*	MLS	43	7	18.5	54
e^*	ANN	22	4.03	22	69
Proposed MoE with ANN experts ($K = 6$)					
A^*	MOE_ANN	29.5	3.7	39	73
e^*	MOE_ANN	17.3	3.1	35	80

Table 7 Overall results

Output	Type	$\hat{E}_{\max}$ (%)	$\hat{E}_{\text{mean}}$ (%)	$\alpha = 1\%$ (%)	$\alpha = 5\%$ (%)
Best surrogate model over the whole domain D					
A^*	ANN-MLS	47	3.5	43.4	80.3
e^*	ANN-MLS	30.5	2.68	42.5	83.3
Mixture of MoE's					
A^*	MOE_ANN	29.5	2.3	60	86.2
e^*	MOE_ANN	19.5	2.1	52	89

(thickness and cross-section area) do not vanish and therefore we do not divide by 0. In addition, in a previous work (Merval 2008) this kind of regression (regression of an optimization results) was already tested with respect to that error measure. We observe that the MoE technique always outperforms the best surrogate built on the whole sub-domain for all error measures. For the whole domain, where the global expert is a simple combination of all local models: MoE and ANN, we observe that the error decreases by about 34% for the section area and by 22% for the thickness.

6 Conclusion

We presented an original method to improve accuracy for regression of the objective and constraints functions that arise in structural optimization problems. Despite the functions to approximate are discontinuous, our method allows to build smooth approximations. Instead of assembling a global surrogate model, this method subdivides the global conjoint space by means of EM algorithm for Gaussian mixture models, designs a surrogate model on each of these sub-regions, combines all the surrogate models using the Gaussian parameters estimated through EM algorithm. This clustering can be hard or fuzzy. We implemented and tested the hard clustering version of this method where local surrogate models can be linear and quadratic regression, moving least squares and artificial neural networks. We also derived a classical surrogate model (weighted least squares in Appendix B) from the soft clustering version. This method is tested over mathematical test cases and an industrial case from aeronautics and is found to increase the accuracy of approximation when compared to a sole surrogate model built over the whole domain, especially in the case of a mixture of ANN's. However, as pointed it out in the first section, the literature on ensemble of surrogate models advocates to mix different surrogate models built over the whole learning basis. As far as smooth approximations of discontinuous functions are concerned, the method presented here needs to be compared with the

different methods of ensembles of surrogate models that can be found in the literature, this will be part of our future research. When smoothness is not required (in case for instance of derivative-free optimization, genetic or evolutionary optimization, ant colony optimization...), the hard mixture version that results in a discontinuous global predictor is more appropriate, this will be also part of our future research. Nonetheless, this method offers many advantages, since the training of the local experts can be performed concurrently and drastically reduces the size of the sample data when the number of clusters gets high. It also offers a very good quality even in the linear case (MOE_LIN) which is inexpensive to compute. It also provides local information on the quality of the regression in the hard-clustering case. Indeed, we know the clusters where the error is high. We also carried out a numerical procedure to find the optimum (or at least an appropriate) number of clusters K. As pointed out, one of the main drawbacks of our proposed algorithm is the large number of data needed for clustering. We could improve our technique by using the local information to enrich the clusters where the approximation is the worst. In addition, the accuracy could be further improved by updating the parameters of the local expert within EM algorithm or use a local procedure to get a better local expert (boosting Meir and Ratsch 2003). Lastly, this algorithm can be adapted to mixed input values (discrete and continuous input variables) by centering clusters on each possible discrete value, making an overall continuous surrogate models over mixed variables. Possible applications towards laminated composite material design and optimization are in progress.

Acknowledgments The authors wish to thank the anonymous reviewers for their insightful and constructive comments. They would like to extend their grateful thanks to L. Jones and J. R. Barron, from Airbus and to A. Shahdin, from ISAE-SupAéro for their careful reading of the manuscript.

Appendix A: EM algorithm

The EM algorithm aims at solving maximum likelihood parametric estimation problems. The EM algorithm was first described in Dempster et al. (1977). First, let us recall the well-known maximum-likelihood estimator. Suppose we have a set of observed data $\mathbf{X} = \{x_i\}_{i=1\ldots N}$ coming from a random variable X. Actually, we shall consider that $\mathbf{X}$ is a size N iid sample, i.e. that it is a realization of N independent identically distributed random elements with the same law than the random variable X. We assume that the law of X is parametric, governed by parameter θ, e.g for a 1D Gaussian variable X, the parameter θ is (μ, σ^2) the mean and variance. We can therefore denote the probability density function $g_\theta(x)$. Now define the likelihood

$$\mathcal{L}(\theta; \mathbf{X}) = \prod_{i=1}^{N} g_\theta(x_i) \tag{23}$$

which is the probability density of the observed data. $\mathcal{L}$ is a function of θ for $\mathbf{X}$ fixed. The sample log-likelihood is

$$l(\theta; \mathbf{X}) = \sum_{i=1}^{N} \log g_\theta(x_i) \tag{24}$$

Maximum likelihood estimator is the value $\hat{\theta}$ of θ which maximizes $l(\theta; \mathbf{X})$. In simple applications, where the law of X belongs to a classical family as Gaussian, Poisson, Gamma, Student, exponential, the maximum-likelihood estimator is simple to compute and amounts to an analytic expression. But in a lot of modern problems, the law of interest is much more complex and can be readily represented by a mixture $\sum_k \alpha_k g(\theta_k, x)$. Notice that this law is the marginal law of X if we add a latent random finite variable K such that the random couple (K, X) is governed by

$$\forall k, \Omega,\ \mathbb{P}(K = k, X \in \Omega) = \alpha_k \int_\Omega g(\theta_k, x) dx$$

Now suppose that the latent variable K is observed, let us note $(\mathbf{K}, \mathbf{X}) = \{(K_i, X_i)_i\}$ this virtual sample and let $\mathbf{X}_k$ be the sub-sample of $\mathbf{X}$ of the data X_i for which the associate latent variable K_i is equal to k, let N_k be the size of $\mathbf{X}_k$. Then it is easy to get the maximum likelihood estimation for θ_k and α_k, namely

$$\hat{\theta}_k = \text{argmax}_\theta l(\theta; \mathbf{X}_k), \qquad \hat{\alpha}_k = \frac{N_k}{N}$$

Actually, we do not know the K_i's whenever they have a physical existence. If the parameter set $\{\theta_k, \alpha_k\}$ was known, K_i could be recovered through a Bayes posterior estimation

$$\gamma_{i,k}(\theta) = \mathbb{P}(K_i = k | \theta, X = X_i) \tag{25}$$

which is called the responsibility of model k for observation i. Indeed, an estimation of the latent variable may be estimated through the MAP (maximum a posteriori) estimation

$$\hat{K}_i = \text{argmax}_k \gamma_{i,k}(\theta)$$

A similar method is used in the proposed technique for ''hard clustering''.

We can now derive the EM algorithm as an iterative relaxation of these two steps. Let us describe the $(n+1)$-th iteration

1. Take the estimates at the previous step $\{(\hat{\theta}_k^n, \hat{\alpha}_k^n)_k\}$
2. Expectation step. Compute associate responsibilities $\hat{\gamma}_{i,k}^n$ for $i = 1 \dots N$ and $k = 1 \dots m$:

$$\hat{\gamma}_{i,k}^n = \frac{\hat{\alpha_k^n} g_{\hat{\theta}_k^n}(x_i)}{\sum_{j=1}^m \alpha_j^n g_{\hat{\theta}_j^n}(x_i)} \tag{26}$$

3. Maximization step. Compute the weighted maximum likelihood estimators for each component of the mixture:

$$\hat{\theta}_k^{n+1} = \frac{\sum_{i=1}^N \gamma_{i,k}^n x_i}{\sum_{i=1}^N \gamma_{i,k}^n} \tag{27}$$

$$\hat{\alpha}_k^{n+1} = \frac{\sum_{i=1}^N \hat{\gamma}_{i,k}^n}{\sum_{j=1}^m \sum_{i=1}^N \hat{\gamma}_{i,j}^n} \tag{28}$$

The convergence of this algorithm is proven in Wu (1983).

Appendix B: An MoE interpretation of weighted least squares

We give here an interpretation of the weighted least squares regression. Say we have $\mathcal{X} = (x_1, \dots, x_n) \in \mathbb{R}^d$ together with their output values $\mathcal{Y} = (y_1, \dots, y_n)$. A local Weighted Least Squares regression at point $\hat{x} \in \mathbb{R}^d$ consists in finding the best multivariate polynomial approximation $f_{\hat{x}}$:

$$f_{\hat{x}} = \text{argmin}_{p \in \prod_s^d} \sum_{i=1}^n \theta(||x_i - \hat{x}||)||p(x_i) - y_i||^2 \tag{29}$$

where Π_s^d is the set of multivariate polynomials of $d-$variables and degree s and θ is a symmetric positive and decreasing function: maximum value at 0 and 0 when $|x| \to +\infty$: which can be strictly positive or compactly-supported. θ is called a weight function. A very popular choice for θ is

$$\theta(r) = \frac{1}{\sqrt{(2\pi)^d \sigma^{2d}}} e^{-\frac{1}{2}\frac{r^2}{\sigma^2}} \tag{30}$$

This regression $f_{\hat{x}}$ is a local approximation which is valid only in the neighborhood of $\hat{x}$. Note that the moving least squares regression consists in a global weighted least squares by making the $\hat{x}$ varying (or moving) over the whole domain $\mathbb{R}^d$, this global model happens to be continuous and even smooth whenever the weight function is smooth (Levin 1998). We depicted several WLS regression on different points together with the MLS regression (Fig. 7). Another way to get a global model is by using a partition of unity. Namely, we have K so called support points $\hat{x}_1, \dots, \hat{x}_K$, a local WLS regression $\hat{f}_{\hat{x}_j} = \hat{f}_j$ is built at each support point $\hat{x}_j$ and the global model is

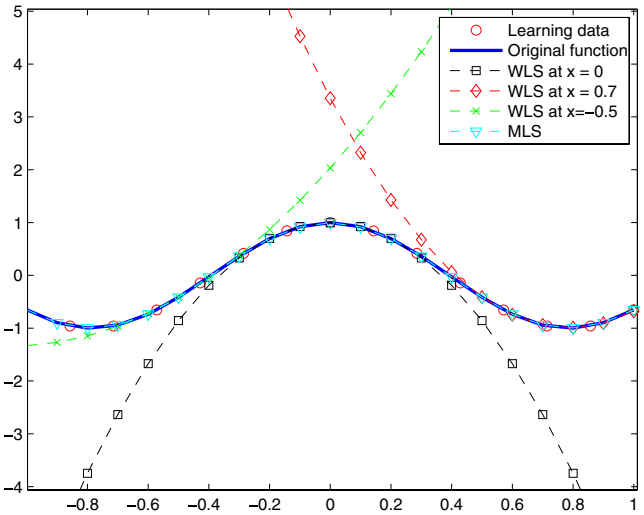


Fig. 7 Weighted least squares regressions (polynomials of degree 2 and $\sigma = 0.1$) of the function $x \to \cos(4x)$ at different points and moving least squares regression

$$F_{WLS}(x) = \sum_{i=1}^K \beta_i(x) \hat{f}_i(x) \tag{31}$$

where

$$\beta_j(x) = \frac{e^{-\frac{1}{2}\frac{||x - x_j||^2}{\sigma^2}}}{\sum_{i=1}^K e^{-\frac{1}{2}\frac{|||x - x_i|}{\sigma^2}}}. \tag{32}$$

This popular model has an obvious interpretation in terms of MoE and soft clustering. Assume that the law[5] of $X \sim \sum_{j=1}^K \frac{1}{K} \mathcal{N}_d(\hat{x}_j, \sigma^2 I_d)$. In that case, we have, keeping the same notations as in Section 2

$$\mathbb{P}(X = x|j) = \frac{1}{\sqrt{(2\pi)^d \sigma^{2d}}} e^{-\frac{1}{2}\frac{||x - \hat{x}_j||^2}{\sigma^2}} \tag{33}$$

[5] In this article we focused on Gaussian mixture models that were fully free, i.e. all the parameters of the Gaussian mixture models are not constrained and EM algorithm estimates all the parameters. There are more simple Gaussian mixture models that assume that all the means are the same or all of the variance-covariance matrices are of the form $\sigma^2 I_n$ (this hypothesis is known in statistics as homoscedascity).

and

$$\mathbb{P}(j|X=x) = \frac{e^{-\frac{1}{2}\frac{||x-\hat{x}_j||^2}{\sigma^2}}}{\sum_{i=1}^{K} e^{-\frac{1}{2}\frac{||x-\hat{x}_i||}{\sigma^2}}} \tag{34}$$

and we see that

$$F_{WLS}(x) = \sum_{i=1}^{K} \mathbb{P}(\kappa = i|X=x)\hat{f}_i(x) \tag{35}$$

and the WLS model is an MoE for $X \sim \sum_{j=1}^{K} \frac{1}{K}\mathcal{N}_d(\hat{x}_j, \sigma^2 I_d)$ using a soft partitioning and where local experts are multivariate polynomial regression experts weighted by the probabilities of the soft partitioning.

Appendix C: Samgen: a sample generator of discontinuous functions for surrogate modeling

When studying regression methods, one often builds surrogate models over closed-form functions to assess the accuracy of the surrogate. We wanted to explore the accuracy of surrogate models on discontinuous functions in high dimension (>8). We ended up with the need of such analytical functions, Samgen was created to generate such functions and more precisely functions that mimic the behavior of aeronautical stress tools. Samgen is a simple Matlab code that generates a structure that contains all the data needed to compute a discontinuous function in arbitrary dimension over $[0, 1]^d$ (Fig. 8). It simply separates with hyper-planes the unit hyper-cube $[0, 1]^d$ into the desired number of discontinuities and over each sub-domain generates two non-convex quadratic functions and compute the maximum of them to generate derivative-discontinuities.

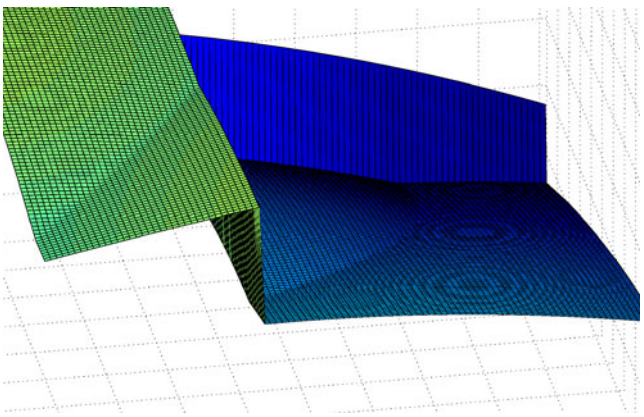


Fig. 8 Example of a $\mathcal{C}^0$ and $\mathcal{C}^1$ discontinuous 2D function created with Samgen

References

Acar E, Rais-Rohani M (2009) Ensemble of metamodels with optimized weight factors. Struct Multidisc Optim 37(3):279−294

Berkhin P (2002) Survey of clustering data mining techniques

Bradley PS, Fayyad U, Reina C (1998) Scaling EM (expectation-maximization) clustering to large databases. Microsoft Research Report, MSR-TR-98-35

Buhmann MD (2001) Radial basis functions. Acta Numer 9:1−38

Burnham KP, Anderson DR (2004) Multimodel inference: understanding AIC and BIC in model selection. Sociol Methods Res 33(2):261

Dempster AP, Laird NM, Rubin DB (1977) Maximum likelihood from incomplete data via the EM algorithm. J R Stat Soc B (Methodological) 39:1−38

Dreyfus G (2005) Neural networks: methodology and applications. Springer, Berlin

Fasshauer GE (2005) Dual bases and discrete reproducing kernels: a unified framework for RBF and MLS approximation. Eng Anal Bound Elem 29(4):313−325

Forrester AIJ, Keane AJ (2009) Recent advances in surrogate-based optimization. Prog Aerosp Sci 45(1−3):50−79

Friedman J, Hastie T, Tibshirani R (2001) The elements of statistical learning

Haftka RT, Gurdal Z (1992) Elements of structural optimization. Kluwer

Haykin S (2008) Neural networks: a comprehensive foundation. Prentice Hall

Jordan MI, Jacobs RA (1994) Hierarchical mixtures of experts and the EM algorithm. Neural Comput 6(2):181−214

Kleijnen JPC, Sanchez SM, Lucas TW, Cioppa TM (2005) A user's guide to the brave new world of designing simulation experiments. INFORMS J Comput 17(3):263−289

Kohavi R (1995) A study of cross-validation and bootstrap for accuracy estimation and model selection. In: International joint conference on artificial intelligence, vol 14. Citeseer, pp 1137−1145

Levin D (1998) The approximation power of moving least-squares. Math Comput 67(224):1517−1532

Liu B, Haftka RT, Watson LT (2004) Global-local structural optimization using response surfaces of local optimization margins. Struct Multidisc Optim 27(5):352−359

Meir R, Ratsch G (2003) An introduction to boosting and leveraging. Lect Notes Comput Sci 2600:118−183

Merval A (2008) Application des modèles réduits à l'optimisation multi-niveau de structures aéronautiques. Thèse SupAéro

Merval A, Samuelides M, Grihon S (2006) Application of response surface methodology to stiffened panel optimization. In: 47th AIAA/ASME/ASCE/AHS/ASC structures, structural dynamics, and materials conference, 1−4 May 2006, Newport, RI

Myers RH, Montgomery DC, Anderson-Cook CM (2009) Response surface methodology, process and product optimization using designed experiments. Wiley

Nealen A, Darmstadt TU (2004) An as-short-as-possible introduction to the least squares, weighted least squares and moving least squares methods for scattered data approximation and interpolation. URL: http://www.nealen.com/projects

Picard RR, Cook RD (1984) Cross-validation of regression models. J Am Stat Assoc 79(387):575−583

Sanchez E, Pintos S, Queipo NV (2008) Toward an optimal ensemble of kernel-based approximations with engineering applications. Struct Multidisc Optim 36(3):247−261

Simpson TW, Toropov V, Balabanov V, Viana FAC (2008) Design and analysis of computer experiments in multidisciplinary design optimization: a review of how far we have come or not. In: Proceedings of the 12th AIAA/ISSMO multidisciplinary analysis and optimization conference, 2008 MAO. Victoria, Canada

Smola AJ, Schölkopf B (2004) A tutorial on support vector regression. Stat Comput 14(3):199−222

van Beers WCM, Kleijnen JPC (2004) Kriging interpolation in simulation: a survey. In: Proceedings of the 36th conference on winter simulation. Winter simulation conference, pp 113−121

Viana FAC, Haftka RT, Steffen V (2009) Multiple surrogates: how cross-validation errors can help us to obtain the best predictor. Struct Multidisc Optim 39(4):439−457

Wang GG, Shan S (2007) Review of metamodeling techniques in support of engineering design optimization. J Mech Des 129:370

Wu CFJ (1983) On the convergence properties of the EM algorithm. Ann Stat 11(1):95−103

Yang Y (2003) Regression with multiple candidate models: selecting or mixing? Stat Sin 13(3):783−810

Zerpa LE, Queipo NV, Pintos S, Salager JL (2005) An optimization methodology of alkaline−surfactant−polymer flooding processes using field scale numerical simulation and multiple surrogates. J Pet Sci Eng 47(3−4):197−208