Available online at www.eccomasproceedia.org
Eccomas Proceedia EUROGEN (2023) 85-113

ECCOMAS
Proceedia

EUROGEN 2023
15th ECCOMAS Thematic Conference on
Evolutionary and Deterministic Methods for Design, Optimization and Control
N. Gauger, K. Giannakoglou, M. Papadrakakis, J. Periaux (eds.)
Chania, Crete Greece, 1–3 June 2023

SCALABLE CLUSTERED ACTIVE SUBSPACES FOR KRIGING IN HIGH DIMENSION

Maxime Chapron[1,2], Christophe Blondeau[1], Michel Bergmann[2], Itham Salah el Din[1], Denis Sipp[1]

[1] ONERA, Université Paris-Saclay, F-92322 Châtillon, France
e-mail: maxime.chapron@onera.fr

[2] INRIA Bordeaux - Memphis Team
351 cours de la Liberation, 33405, Talence, France
e-mail: michel.bergmann@inria.fr

Abstract. *Useful parameterisations of shapes for engineering models often climb from many tens to a few hundred design variables, potentially jeopardising usual optimisation techniques. Surrogate models are intractable when parameter numbers exceed a few tens, and gradient descent through adjoint computation in high-dimension is threatened by potential multimodality. Dimension reduction addresses these problems nicely. This paper aims to improve upon the Active Subspaces dimension reduction method by applying a combination of active subspaces in subregions of the design space, where their use of the objective function's gradients will exploit useful local information to discover specific trends instead of trying to identify global trends over an entire design space. The partitioning of the input space is done through Gaussian Mixture Model clustering, where the authors assume the distribution of the joint inputs/outputs to be a mixture of Gaussians. This GMM clustering is used to drive a Support Vector Machines (SVM) based supervised classifier, which will enable the definition of the overlapping zones in the input space. The use of overlapping increases prediction accuracy at the boundaries between clusters, an improvement over recombination methods typically found in the literature. The local expert of choice is Kriging, in part due to its built-in variance prediction.*

Keywords: Dimension Reduction, Active Subspaces, Machine Learning, Surrogate Modeling, Kriging

Peer-review under responsibility of the organizing committee of EUROGEN 2023.
doi: 10.7712/140123.10192.18902

1 INTRODUCTION

Aerodynamic shape optimisation plays a fundamental role in aircraft design. Modern aerospace design relies heavily on computer simulations to circumvent expensive and time-consuming physical experiments. Yet, as design complexity continues to increase, the computational cost for these numerical simulations becomes uncomfortably high, even on large scale computing clusters. The problem is exacerbated in cases where the simulation needs to be queried multiple times. Optimisation is one of these cases, since minimising an objective function with regards to input parameters requires repeated evaluations of successive designs. Uncertainty quantification (UQ) is another, where simulations must be run repeatedly in order to evaluate the influence of variability in the input parameters. In optimisation, we typically seek to minimise a scalar quantity of interest (QoI) such as drag or lift-to-drag ratio for winged-typed surfaces (under lift and/or moment constraints) or isentropic efficiency (under pressure ratio and mass flow requirements) for engine components. However, useful parameterisations of shapes for engineering models often result in high-dimensional design spaces (several tens to several hundreds of design variables) which can create challenges for both local and global optimisers. A workaround to successive evaluations is surrogate modelling. A statistical model of the function of interest is trained using an initial set of simulations. This initial investment is offset by using the model in lieu of actual CFD computations in order to evaluate the performance of new designs, since calls to the surrogate model are essentially free.

The considered functions of interest have been shown to be potentially multimodal, either due to non-convexity of the QoI with respect to the parameter space [1], or poorly converged gradient computation [2], and as dimensionality rises we expect function topology to keep increasing in complexity. When facing tortuous, possibly multimodal functions, a common practice is to cluster the design space, essentially splitting up the problem into several smaller, simpler problems. This Divide and Conquer approach is beneficial in the case of surrogate modelling, since each surrogate model is now tasked with approximating the function in a subregion of the design space, where it is hoped the objective function has less complex local variations.

A major limitation of typical response surfaces such as Kriging is that they do not easily scale to high dimensions. This problem, amongst others, can be addressed nicely through dimension reduction. A technique for reducing the dimension of the parameter space has been proposed by Constantine [3]. The idea consists in revealing a low-dimensional subspace that captures the trends in the QoIs by leveraging gradient information, and then exploits these directions to efficiently find an optimal design in the appropriate areas of the design space. This subspace is called the Active Subspace (AS). The method is justified since many engineering QoIs exhibit monotonic behaviour with respect to the input parameters. It is rooted in the same principles as well known Principal Component Analysis (PCA) or Proper Orthogonal Decomposition (POD) methods but, unlike the two aforementioned methods who only discover correlation in the input or output spaces, the use of gradient information enables the identification of relationships between the input and output spaces. When compared to the PLS method [4], the Active Subspaces method benefits from the use of gradient information, as well as introducing a probabilistic framework which allows for built-in estimation of the quality of the spectral information.

In this paper, we present a Clustered Active Subspaces strategy that combines a model-based clustering using Gaussian mixture with local Active Subspace discovery. Kriging surrogate models are then constructed in each of the reduced subspaces. These local experts are appro-

priately recombined to obtain a scalable surrogate model for a high-dimensional design space. Typically, in the case of highly complex topologies, a mixture of experts strategy is expected to improve model accuracy. However, with the smooth recombination using a weighted sum of neighbouring experts, this may not always be the case: early tests have shown the bulk of the error being located at the cluster boundaries. We attribute this result to Kriging's (and other response surfaces) poor performance when extrapolating. We propose an original overlapping-based approach to circumvent this difficulty, whereby the training sets of the response surfaces are enhanced by including appropriate points from nearby clusters. In doing so, we create an overlapped area around the cluster boundaries, where both experts are more likely to be accurate since they have been trained on surrounding points. The use of gradient information is a recurring theme in this work as we want to benefit from this additional information, since have access to the adjoint solver embarked in the ONERA elsA CFD solver [5], which yields the gradients at an affordable cost [6].

Figure 1: Clustered Active Subspaces Strategy

We demonstrate the efficiency of this approach on the Branin analytical function [7] and on two existing datasets for the 2D NACA12 aerofoil and the 3D ONERA M6 wing [8]. These datasets are available as part of Constantine's Active Subspaces toolbox for Python.

The paper is structured as follows: we begin by explaining the strategy behind the Clustered Active Subspaces method, before laying out the groundwork for Active Subspaces, Gaussian Mixture Model clustering and Kriging. Section 3 explains the two improved recombination strategies developed here, and section 4 presents the novel automatic dimension selection criterion for AS. Section 5.1 introduces the two physics-based datasets used to test our method, and section 5 provides a performance analysis for the CAS method on the aforementioned test cases. We will close with a discussion of the results.

2 METHODOLOGY

We now present the building blocks of the Clustered Active Subspace method. A presentation of the method's workflow is presented in section 2.4

2.1 Kriging

Kriging is a family of metamodels stemming from Gaussian Processes, benefiting from embedded uncertainty modelling. Like other response surfaces, the objective is to learn the relationship between a numerical experiment's inputs and outputs, in order to predict new values without the need for new, expensive simulations.

The distribution of a Gaussian process is fully characterised by:

- its mean function $\boldsymbol{\mu}$ defined over $\mathcal{X} \subset \mathbb{R}^d$
- its covariance function, kernel $k\left(.,.\right)$, defined over $\mathcal{X}\times\mathcal{X} : k(\mathbf{x},\mathbf{x}') = \text{cov}\left(Y(\mathbf{x}), Y(\mathbf{x}')\right)$

Note the input is the d dimensional vector $\mathbf{x} \in \mathcal{X}$. In this work, we consider without any loss of generality the quantity of interest to be a scalar output function $Y(\mathbf{x}) \in \mathbb{R}$.

The current statistical model can thus be expressed as follows:

$$\begin{aligned} &Y(\mathbf{x}) \sim \mathcal{N}\big(\boldsymbol{\mu}(\mathbf{x}), k(\mathbf{x},\mathbf{x})\big) \\ &Y(\mathbf{x}) = \boldsymbol{\mu}(\mathbf{x}) + Z(\mathbf{x}) \qquad \boldsymbol{\mu} \text{ is known (deterministic)} \\ &Z(\mathbf{x}) \sim \mathcal{N}\left(0, k(\mathbf{x},\mathbf{x})\right) \end{aligned} \tag{1}$$

A valid kernel must be symmetric ($k(\mathbf{u},\mathbf{u}') = k(\mathbf{u}',\mathbf{u})$) and positive semi-definite ($\forall \mathbf{u} \in \mathbb{R}^M, \mathbf{u}^T\mathbf{K}\mathbf{u} \geq 0, \mathbf{K} = k(\mathbf{u},\mathbf{u}), \mathbf{K} \in \mathbb{R}^{M\times M}$). Although many functions satisfy these criteria, in this work we will use exclusively the squared exponential kernel, whose full expression is given below.

$$k(\mathbf{x},\mathbf{x}') = \sigma^2 \exp\left(-\frac{1}{2}\sum_{i=1}^{d}\frac{(\mathbf{x}_i - \mathbf{x}_i')^2}{\theta_i^2}\right), \qquad k : (\mathbf{x},\mathbf{x}') \to k(\mathbf{x},\mathbf{x}'), \quad \mathbb{R}^{d\times d} \to \mathbb{R} \tag{2}$$

With Gaussian vector conditioning, we can write the joint distribution of the observed target value and the function values at the test location $\mathbf{x}_*$ under the prior:

$$\begin{bmatrix}\mathbf{y}\\ \mathbf{y}_*\end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} k(\mathbf{X},\mathbf{X}) + \sigma_n^2 I & k(\mathbf{X},\mathbf{x}_*) \\ k(\mathbf{x}_*,\mathbf{X}) & k(\mathbf{x}_*,\mathbf{x}_*)\end{bmatrix}\right), \qquad \mathbf{X} \in \mathbb{R}^{M\times d} \tag{3}$$

from which we arrive at the predictive equations for regression Kriging:

$$\hat{\boldsymbol{\mu}}(\mathbf{x}_*) = \boldsymbol{\mu} + k(\mathbf{x}_*,\mathbf{X})\left[k(\mathbf{X},\mathbf{X}) + \sigma_n^2\mathbf{I}\right]^{-1}(\mathbf{y} - \boldsymbol{\mu}) \tag{4}$$

$$\hat{\sigma}^2(\mathbf{x}_*) = \hat{\sigma}^2 - k(\mathbf{x}_*,\mathbf{X})\,k(\mathbf{X},\mathbf{X})^{-1}\,k(\mathbf{x}_*,\mathbf{X})^T \tag{5}$$

The Kriging hyperparameters are determined by maximizing the probability of the hyperparameters, given the observations. This can be done in of two ways, either through optimisation of Maximum Likelihood function [9], or through Cross-Validation. In either case, as dimensionality rises, complexity rises to the point where the numerical problem becomes intractable. This is where dimension reduction comes into play.

2.2 The Active Subspace method

2.2.1 The basis behind the method

Accurate gradient information is available, so we use the Active Subspaces method proposed by Constantine [3].

Consider a function f with m continuous inputs. The column vector $\mathbf{x}$ takes values in $\mathbb{R}^d$; we write $\mathbf{x} = [x_1, \dots x_d]$.

$$f = f(\mathbf{x}), \quad \mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^d \tag{6}$$

Let $\mathcal{X}$ be equipped with a bounded probability density function $\rho : \mathbb{R}^d \to \mathbb{R}_+$, where

$$\rho(\mathbf{x}) > 0, \mathbf{x} \in \mathcal{X} \tag{7}$$

Assuming f is continuous and square-integrable with respect to ρ, and with the gradient of f stored in the column vector $\nabla_{\mathbf{x}} f(\mathbf{x}) = \left[\frac{\partial f}{\partial x_1} \cdots \frac{\partial f}{\partial x_d}\right]^T$, we define the $d \times d$ correlation matrix $\mathbf{C}$ as the expectation of the outer product of the gradient vector with itself:

$$\mathbf{C} = \mathbb{E}_\rho\left[(\nabla_{\mathbf{x}} f)(\nabla_{\mathbf{x}} f)^T\right] \tag{8}$$

Since $\mathbf{C}$ is symmetric and positive semidefinite, consider its eigenvalue decomposition:

$$\mathbf{C} = \mathbf{W}\mathbf{\Lambda}\mathbf{W}^T, \quad \mathbf{\Lambda} = \text{diag}(\lambda_1, \ldots, \lambda_d), \lambda_1 \geq \cdots \geq \lambda_d \geq 0, \quad \mathbf{W}^T\mathbf{W} = \mathbf{I} \tag{9}$$

It is shown that the gradient of f with respect to an eigenvector is equal to the corresponding eigenvalue.

Lemma 2.1. *The mean-squared directional derivative of f with respect to the eigenvector $\mathbf{w}_i$ is equal to the corresponding eigenvalue:* $\mathbb{E}\left[\left((\nabla_{\mathbf{x}} f)^T f \mathbf{w}_i\right)^2\right] = \lambda_i$.

With the eigenvalues sorted in decreasing order, we are able to separate eigenvectors (columns of $\mathbf{W}$) into two sets.

$$\mathbf{\Lambda} = \begin{bmatrix} \mathbf{\Lambda}_1 & \\ & \mathbf{\Lambda}_2 \end{bmatrix}, \qquad \mathbf{W} = [\mathbf{W}_1 \quad \mathbf{W}_2] \tag{10}$$

where $\mathbf{\Lambda}_1 = \text{diag}(\lambda_1, \ldots, \lambda_n)$ with $n < d$, and $\mathbf{W}_1$ is $d \times n$.

Define the new, partitioned variables $\mathbf{y} \in \mathbb{R}^n$ and $\mathbf{z} \in \mathbb{R}^{d-n}$ by

$$\mathbf{y} = \mathbf{W}_1^T\mathbf{x}, \quad \mathbf{z} = \mathbf{W}_2^T\mathbf{x} \tag{11}$$

from which any point of the original domain $\mathcal{X}$ can be expressed:

$$f(\mathbf{x}) = f(\mathbf{W}_1\mathbf{y} + \mathbf{W}_2\mathbf{z}) \tag{12}$$

Consider the following lemma:

Lemma 2.2. *The mean-squared gradients of f with respect to the coordinates $\mathbf{y}$ and $\mathbf{z}$ satisfy:*

$$\mathbb{E}\left[(\nabla_{\mathbf{y}} f)(\nabla_{\mathbf{y}} f)^T\right] = \lambda_1 + \ldots + \lambda_n, \tag{13}$$

$$\mathbb{E}\left[(\nabla_{\mathbf{z}} f)(\nabla_{\mathbf{z}} f)^T\right] = \lambda_n + \ldots + \lambda_d \tag{14}$$

Proofs for both of these Lemmas are found in [10]. With this split, we know that, *on average*, $f(\mathbf{x})$ is more sensitive to perturbations along the first set of eigenvectors. This result justifies the term *active subspace*. If, for example, the smallest eigenvalue λ_d is exactly zero, the mean-squared change in f along $\mathbf{w}_d$ is null. Therefore, discarding the d^{th} direction would lead to exact dimension reduction, where no information is lost. This is the ideal scenario.

2.2.2 Approximation in the active subspace

Our goal is to construct a function $g = g(\mathbf{y})$ that depends only on the $n < d$ active variables, and study its parametric dependence as a proxy for $f(\mathbf{x})$, since we assume that the number of parameters d is too large for proper standard response surface construction. Essentially, we wish to approximate an d-variate function f by a function which is $\mathbf{z}$-invariant. Doing so necessitates managing at least two approximations. First, approximating f by a $\mathbf{z}$-invariant function incurs information loss, which leads to errors. Secondly, we must build an n-variate response surface, which, by construction, may not always be perfectly fitted. These errors are bounded, and they are analysed in this section.

Conditional expectation For a certain coordinate $\mathbf{y}$, the best guess one can make at the value of f is the average over all values of $\mathbf{x}$ that map to $\mathbf{y}$.

$$g(\mathbf{y}) = \mathbb{E}\left[f \mid \mathbf{y}\right] = \int_{\mathbf{z}} f\left(\mathbf{W}_1\mathbf{y} + \mathbf{W}_2\mathbf{z}\right) \pi_{Z|Y}(\mathbf{z})\, d\mathbf{z} \tag{15}$$

Being a conditional expectation, G is the best mean-squared approximation of f given $\mathbf{y}$.

As such, we can approximate the original function $f(\mathbf{x})$ with a function, g, defined in the reduced space:

$$f(\mathbf{x}) \approx F(\mathbf{x}) \equiv g(\mathbf{W}_1^T\mathbf{x}) \tag{16}$$

Doing so will inevitably lead to errors, unless the eigenvalues of the discarded subspace $\mathbf{W}_2$ are zero (exploiting Lemma 2.2). Further, theorem 2.3 provides an upper bound to the error commited by this approximation.

Theorem 2.3. *The mean squared error of F defined in (16) is bounded from above by:* $\mathbb{E}\left[(f-F)^2\right] \leq C_1(\lambda_{n+1} + \ldots + \lambda_d)$*, with* C_1 *a constant.*

The trouble with the approximation of g is that each evaluation of it requires an integration with regards to the discarded variables $\mathbf{z}$. In the case of successful active subspace discovery, the number of discarded directions can be high, leading to consequential numerical integration costs. However, if the discarded eigenvalues $\lambda_{n+1}, \ldots, \lambda_d$ are small, which we hope they are, then small changes in $\mathbf{z}$ produce very little change in f. Thus, the variance of f along $\mathbf{z}$ is small, and a standard Monte Carlo will accurately approximate the conditional expectation g while deriving an error bound on said approximation. Define the Monte Carlo estimate $\hat{g} = \hat{g}(\mathbf{y})$

$$f(\mathbf{x}) \approx \hat{F}(\mathbf{x}) \equiv \hat{g}(\mathbf{W}_1^T\mathbf{x}) \tag{17}$$

which, after combining the usual error bound for a Monte Carlo approximation with the result of Theorem 2.3 yields the total error bound made when approximating a function f by an N-point Monte Carlo estimation of its conditional expectation at a given $\mathbf{y}$.

$$\mathbb{E}\left[(f-\hat{F})^2\right] \leq C_1\left(1 + \frac{1}{N}\right)(\lambda_{n+1} + \ldots + \lambda_d) \tag{18}$$

This gives an easy criterion to evaluate the precision of the subspace. In practice, a single point Monte Carlo is accurate enough to, provided the sum of discarded eigenvalues is small enough. If the error committed by these two successive approximations is too high, we can increase the size of the subspace in order to reduce the error. The other option is to increase the size of the training data set.

2.3 Clustering through Gaussian Mixture Models

Following a Divide and Conquer approach, we wish to split up our design space into smaller domains where the function exhibits a certain trend. Since it is not in advance known what we are looking for, we use unsupervised learning to identify patterns in the data. We formulate a probabilistic model which postulates certain unobserved variables -called latent variables- which correspond to things we are interested in inferring. We are focused on one type of latent variable model: the Gaussian Mixture Model. In complex cases, distributions rarely follow simple distributions. For instance, the data might be multimodal (figure 2).

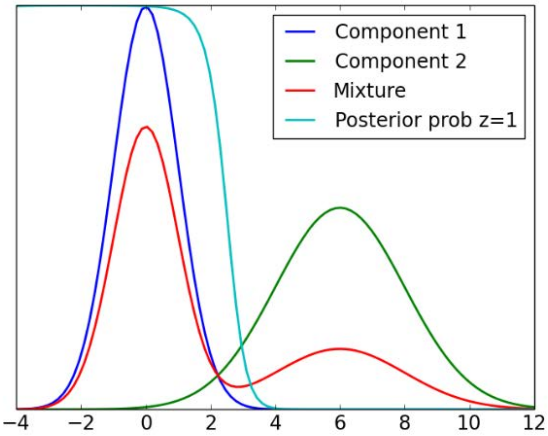


Figure 2: Probability density function of a one-dimensional GMM

In this situation, we model the data in terms of a mixture of several components, where each component has a simple parametric form: a Gaussian distribution. We assume each data point belongs to one of the components, and we try to infer the distribution for each component separately. In contrast to usual methods such as K-means which rely on Euclidean distance between the input locations, our clustering depends on the Mahalanobis distance in the joint input-output space. Our clustering is thus driven by function topology, and is unbothered by space-filling Designs of Experiments, where Euclidean distance is non informative. By considering the joint law of the inputs and outputs (X, Y), function values play a key role in splitting up the domain into clusters where the objective function exhibits similar trends. Note that we could have considered the joint law of the inputs and the gradients in the same manner, and a comparison of both these methods will be carried out in the future.

We are interested in making decisions based on function topology. Therefore, instead of considering only the inputs $\mathcal{X} = (x_1, \dots, x_d)$ where $x_i \in \mathbb{R}^d$ are assumed to come from a set of identical and independently distributed random variables, we consider the joint law of the inputs and their corresponding outputs, $\mathcal{Y} \in \mathbb{R}$. In this work, we only consider the scalar function $y = f(\mathbf{x})$ as the output, so the conjoint space of $(\mathcal{X}, \mathcal{Y}) = \mathcal{Z} = (z_1, \dots, z_N)$ where $\mathbf{z}_i = (\mathbf{x}_i, y_i) \in \mathbb{R}^{d+1}$ is considered. For future work, note that the formalism detailed here would apply in the multi-output case with very little change. In the case of GMM, we assume the probability density function of Z follows a weighted combination of a given number K of multivariate Gaussian laws in $\mathbb{R}^{d+1}$ such that:

$$Z \sim \sum_{k=1}^{K} \alpha_k \, \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{19}$$

where α_k is the proportion of the Gaussian k in the mixture. In eq.19, being probability density functions, it goes without saying that $\forall k \in [1, K], \alpha_k \in [0, 1]$ and $\sum_{k=1}^{K} \alpha_k = 1$. Denote $\boldsymbol{\mu}_k = \begin{pmatrix} \boldsymbol{\mu}_k^X \\ \boldsymbol{\mu}_k^Y \end{pmatrix}$ the X and Y components of the vector of means. Similarly, $\boldsymbol{\Sigma}_k = \begin{pmatrix} \boldsymbol{\Sigma}_k^X & \sigma_k \\ \sigma_k^T & \xi_k \end{pmatrix}$ is the covariance matrix.

Recall that a multivariate Gaussian distribution is defined by its mean $\boldsymbol{\mu}_k$ and its covariance matrix $\boldsymbol{\Sigma}_k \in \mathcal{M}_{d+1}(\mathbb{R})$, which is symmetric positive definite. These parameters are estimated

by the alternating Expectation-Maximisation algorithm [11], which was designed for this purpose.

Once the parameters have been estimated, we can calculate the probability of membership of every point to every cluster, that is to say the probability for a given (x, y) to belong to cluster k. It is given computing the posterior probabilities obtained by Bayes' formula where κ denotes the discrete random variable associated with the clusters:

$$\begin{aligned}\pi\big(\kappa = k \mid (X,Y) = (x,y)\big) &= \frac{\pi(\kappa = k)\ \pi\big((X,Y) = (x,y) \mid \kappa = k_i\big)}{\sum_{k=1}^{K} \pi(\kappa = k)\ \pi\big((X,Y) = (x,y) \mid \kappa = k\big)} \\ &= \frac{\alpha_k\, \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\pi(X,Y)}\end{aligned} \tag{20}$$

We then partition our data points by assigning each point to the cluster to which it has the highest probability of membership. Mathematically, given (x_i, y_i) in the conjoint space, that particular training point i lies in cluster j, where:

$$j^* = \underset{j=1,\dots,K}{\arg\max}\ \pi\left(\kappa = j \mid (X,Y) = (x_i, y_i)\right), \quad i = 1, \dots, N \tag{21}$$

2.3.1 Recombination

Now that the parameters have been identified, and that every point in the dataset has been labelled, we can train the local Kriging experts. Each expert is trained using the data points of that specific cluster. In order to have a global surrogate model able to predict the unknown response y of a new entry $\mathbf{x} \in \mathbb{R}^d$, we must then recombine the local experts, f_k. We proceed by constructing a linear combination of the local experts:

$$\hat{f}(\mathbf{x}) = \sum_{i=1}^{K} \beta_k(\mathbf{x}) f_k(\mathbf{x}) \tag{22}$$

where $\beta = (\beta_1, \dots, \beta_K)$ is the vector of the local weighting functions (local in the sense that its values depend on $\mathbf{x}$). Please note however, that while each expert has been trained using the training set of its specific cluster, each expert is defined over the entire design space. Mathematically, this reads:

The support of each expert $f_k(\mathbf{x})$ is the entire domain $X \in \mathcal{X} \subset \mathbb{R}^d$
The training set of cluster k is denoted X_k
The entire dataset is made up of the union of the training sets, such that:
$X = \cup_{k=1}^{K} X_k$
$X_i \cap X_j = \emptyset, \quad i \neq j$

The most basic form of recombination is what is sometimes known as a *hard recombination*: the gating network is the Heaviside function: $\beta_k(\mathbf{x}) = \begin{cases} 1, \text{ if } \mathbf{x} \text{ belongs to cluster } k \\ 0, \text{ otherwise} \end{cases}$.
This is justified since the local expert f_k is best equipped to accurately predict a value located in cluster k. This kind of recombination is very useful to deal with discontinuous objective functions. However, when dealing with continuous functions, such a recombination strategy

could lead to approximation errors. Furthermore, in the context of optimisation through gradient descent, we need the surrogate model to be continuous and even differentiable, requiring what is most often known as *smooth recombination*.

In order to maximize performance, the weights making up the linear combination will not be constant throughout the design space. Most response surfaces are most accurate when interpolating. Take cluster k, for example. Near $\boldsymbol{\mu}_k$ (the center of cluster k), it is most likely that f_k can provide the most accurate estimation of the objective function, since it has been trained by the surrounding points. Therefore, we will strongly lean on this expert by weighting it heavily. To achieve this, we weigh the experts' predictions according to the location's probability of belonging to a particular cluster. It just so happens that the GMM parameter estimation yields the analytical formula for the probability of belonging to each cluster, $\beta_k = \Big(\pi(\kappa = 1 \mid X = x), \dots \pi(\kappa = K \mid X = x)\Big)$. This leads to the classical expression for mixture of experts smooth recombination, as expressed by Jordan and Jacobs [12]:

$$\hat{f}(\mathbf{x}) = \sum_{k=1}^{K} \beta_k(\mathbf{x}) f_k(\mathbf{x}) = \sum_{k=1}^{K} \pi(\kappa = k \mid X = x) f_k(\mathbf{x}) \tag{23}$$

Notice how the expression of β_k depends only upon $\mathbf{x}$. This result is critical as it allows the surrogate model to predict new points, not just the points at which it was trained. These expressions are easily obtained from the Gaussian parameters computed by the EM algorithm: from the conjoint law $(X, Y) \sim \sum_{k=1}^{K} \alpha_k \mathcal{N}(\mu_k, \Sigma_k)$ we can derive the law of $X \mid \kappa = k$ without knowing y.

$$\pi(X \mid \kappa = k) = \mathcal{N}\big(\boldsymbol{\mu}_k^X, \boldsymbol{\Sigma}_k^X\big) \tag{24}$$

where $\boldsymbol{\mu}_k^X, \boldsymbol{\Sigma}_k^X$ are defined under equation (19). This is different from the laws we would have obtained by applying the EM solely on the inputs: in essence, it is a marginalisation of the probability of membership of the conjoint law, where we have marginalised with regards to the known outputs $\mathbf{y} = \{f(\mathbf{x}_i)\}_{i=1,\dots,N}$.

2.4 Clustered Active Subspaces method

These building blocks enable us to present the strategy behind the Clustered Active Subspaces method.

Algorithm 1: Clustered Active Subspaces method

1 Construct an initial dataset of N samples $\{\mathcal{D}\} = (\mathbf{x}_i, y_i, \nabla_{\mathbf{x}} y_i)$. These are independent samples "drawn" according to the spatial density distribution.

2 Assemble the joint data $\mathbf{z}_i = (\mathbf{x}_i, y_i) \in \mathbb{R}^{d+1}, \quad i = 1, \dots, N$ and set a number of clusters K.

3 Assume $Z = (\mathbf{z}_1, \dots, \mathbf{z}_N)$ follows a Gaussian mixture

$$\pi\left(\mathbf{z} \mid \alpha_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right) = \sum_{k=1}^{K} \alpha_k \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \text{ with } \begin{cases} \boldsymbol{\mu}_k \in \mathbb{R}^{d+1}, \boldsymbol{\Sigma}_k \in \mathcal{M}_{d+1}(\mathbb{R}) \\ \sum_{k=1}^{K} \alpha_k = 1 \end{cases}$$

4 Identify hyperparameters using the Expectation-Maximisation algorithm

5 Perform hard clustering from highest posterior probability. For $(\mathbf{x}_i, y_i) \in \mathbf{z}$:

$$j^* = \arg \max_{j=1,\dots,K} p\big(k = j \mid (X, Y) = (\mathbf{x}_i, y_i)\big) \quad i = 1, \dots, N$$

6 **for** k *in* $[1, K]$ **do**

7 Obtain local empirical covariance of gradients with

$$\mathbf{C}_k \approx \hat{\mathbf{C}}_k = \mathbb{E}_{\rho_k(\mathbf{x})}\left[\nabla f(\mathbf{x}) \nabla f(\mathbf{x})^T\right] = \frac{1}{M} \sum_{j=1}^{M} \nabla f(\mathbf{x}_j) \nabla f(\mathbf{x}_j)^T, \qquad M = |k|$$

8 Decompose matrix $\hat{\mathbf{C}}_k = \mathbf{W}_k \Lambda_k \mathbf{W}_k^T \quad$ with $\mathbf{W}_k^T \mathbf{W}_k = \mathbf{I}_k$

9 In each cluster, $f(\mathbf{x}) \approx G_k(\mathbf{W}_{1,k}^T \mathbf{x})$ with $\mathbf{W}_k = [\mathbf{W}_{1,k}, \mathbf{W}_{2,k}]$

10 Define local AS kernel $k_k(\mathbf{W}_{1,k}^T \mathbf{x}, \mathbf{W}_{1,k}^T \mathbf{x}'; \Theta) \quad$ with $\Theta = (\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$

11 Train local Kriging $f_k(\mathbf{x}_k)$ using the points of current cluster.

12 **end**

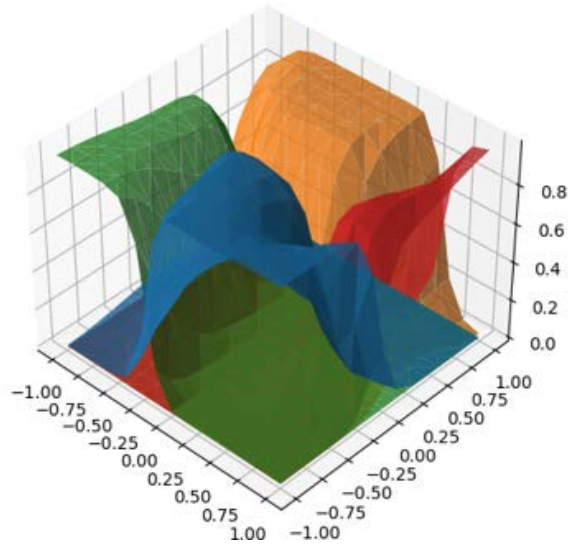
13 Recombine local experts to form a global surrogate model.

3 OVERLAPPING

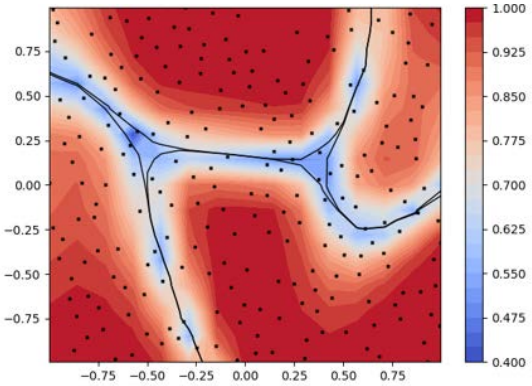
Initial tests have shown that most of the prediction errors of our surrogate model originate at the boundaries between clusters. We attribute this error to extrapolation: each of the local experts are trained using data belonging to their cluster. When asked to predict new values near the boundaries, none of the local response surfaces have been trained in that area, which leads to prediction errors. In order to mitigate this issue, we extend the training set of each cluster so that they encompass these frontier areas, reducing areas of extrapolation and therefore improving predictive accuracy.

3.1 Probability of membership based overlapping

The first idea is to rely more heavily on the probability of membership functions ($\beta_k = \pi(\kappa = k \mid X = x)$) yielded by the Gaussian Mixture Model. These functions are each defined over the entire domain, as shown in figure 3a on a two dimensional space corresponding to the domain of the Branin function [7].



(a) Probability of membership to each cluster in a two-dimensional input space



(b) Maximal probability of membership to a cluster in a two-dimensional input space

Figure 3: Exploiting the probability of membership functions

For each cluster, instead of only assigning points through their maximal probability of membership, we add points whose joint probability of membership is superior to a certain threshold, say $\gamma \geq 0.3$. This threshold could be another hyperparameter we would need to optimally choose. Including too many points would increase computation time as well as increase the possibility of adding irrelevant or even detrimental points to the training set. However, enough points must be added in order to sufficiently enrich the response surface and increase performance by reducing the recombination error around the cluster boundaries.

Figure 4 shows which of the training points of the original dataset are added to the training set of cluster 4 (bottom right cluster). Graphically, white points are the points which have been identified as belonging to cluster 4 by the true probability of membership, as computed by the joint law of (X, Y): $\arg\max_{j=1,\dots,K} \pi\big(\kappa = j \mid (X, Y) = (x_i, y_i)\big)$. However, The marginal probability of membership $\beta_k(\mathbf{x}) = \pi(\kappa = k \mid X = x)$ misclassifies some of these points (black points circled in pink).

Results are promising, since the overlapping procedure identifies both points which have been misclassified and points belonging to other clusters but near the boundaries of cluster 4. As a bonus, all suggested points are pertinent: no points are far away from the cluster, they should all contribute positively to predictive power.

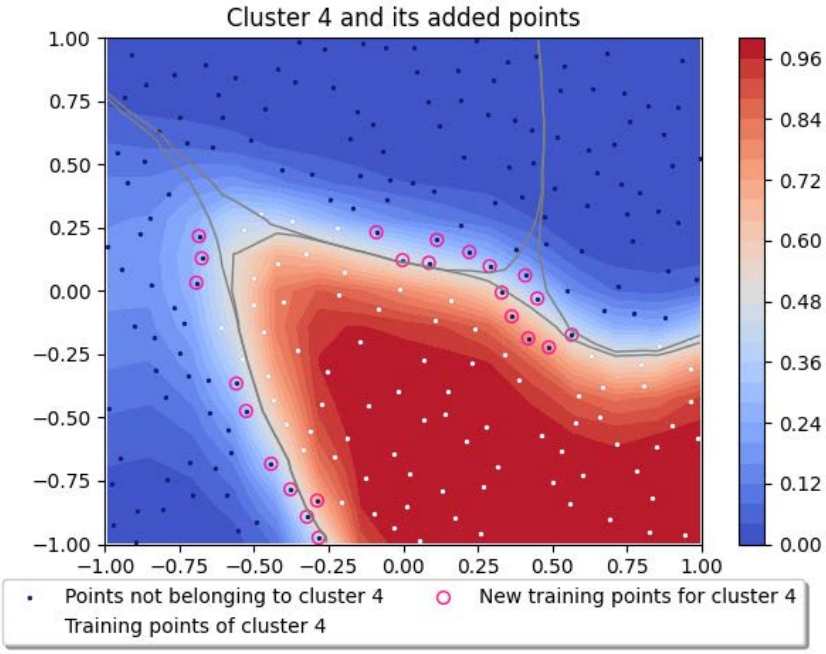


Figure 4: Adding points to a cluster's training set. White points belong to the considered cluster, as determined by the GMM clustering. The contour is the marginal probability of membership to the current cluster, as computed with β_k. Cluster boundaries correspond to $\beta_k = 0.5$. Black points circled in purple are points added to the current cluster's training set by the overlapping procedure.

This strategy is satisfying but only factors in the probability function value. It does not allow us to control the distance (in the parameter space) between the point and the boundary. Depending on the profile of this probability of membership function, additional points can be selected regardless of their distance to the boundary. In order to relieve this limitation, we propose to formulate a criterion which is linked to the Euclidean distance to the boundary. A tool for this kind of job is Support Vector Classifier.

3.2 Supervised learning

We now propose to resort to supervised learning by training a classifier on the labels obtained after GMM clustering. From the gallery of options [13], we chose a Support Vector Machines-based classifier in order to benefit from one of its major advantages, the optimal separation margin. In addition to giving us an explicit formulation for the cluster boundaries, the margin allows us to define a Euclidean distance from each point in the domain to any cluster boundary. This is information we can use to define the appropriate width of the overlapping region.

3.2.1 Support Vector Classifier

Support Vector Machines (SVM) [14] aim to identify the optimal hyperplane which maximises the minimal distance (in the direction normal to said hyperplane) between two points of different sets. Consider our data $(\mathbf{x}_i, y_i)_{i=1,\dots,N}$ where $y_i \in \{-1, 1\}$ are the labels given by a classifier. In the case of linearly separable data, we aim to build a function, the *decision function*

of the form

$$f(x) = \mathbf{w}^T\mathbf{x} + b = 0, \qquad \mathbf{w} \in \mathbb{R}^d \tag{25}$$

Classification is induced by the sign of f, $G(\mathbf{x}) = \text{sign}\left[\mathbf{w}^T\mathbf{x} + b\right]$. The margin between two sets is defined as the minimum distance between all possible distances $\|.\|_{\mathbf{w},b}$ between two points from distinct sets.

$$M(X^1, X^{-1}; \mathbf{w}, b) = \min_{\mathbf{x}^+ \in X^1, \mathbf{x}^- \in X^{-1}} \|\mathbf{x}^+ - \mathbf{x}^-\|_{\mathbf{w},b} \tag{26}$$

The Optimal Separating Hyperplane is defined as the linear separator which maximises the margin:

$$\max_{\mathbf{w},\, b} M(X^1, X^{-1}; \mathbf{w}, b) \tag{27}$$

It is shown in [15] that this maximisation problem simplifies to:

$$\begin{aligned} &\min_{(\mathbf{w},b)\in\mathbb{R}^{d+1}} \|\mathbf{w}\|_2^2 \\ &\text{s.t.} \quad y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 \qquad y_i \in \{-1, 1\} \end{aligned} \tag{28}$$

Another valuable feature of SVMs is that they can be extended to deal with non-linearly separable data, by instead considering:

$$f(x) = \mathbf{w}^T\phi(\mathbf{x}) + b \tag{29}$$

This is known as the kernel trick, where $\phi(.)$ is a non-linear mapping. This is equivalent to artificially increasing the dimension of the parameter space. Using a kernel $K(x_i, x_j) = < \phi(x_i), \phi(x_j) >_H$ does not change the optimization problem, and we do not need to explicitly know the mapping, only its dot product $K(.,.)$. This trick allows for much better classification in the case of data which is tough to separate linearly, as shown in section 5.2.

SVC is traditionally a binary classifier, but extension to the multiclass scenario is straightforward.

4 SELECTION OF THE SIZE OF THE ACTIVE SUBSPACES

One of the key problems in dimension reduction is just how much reduction you can get away with. Inevitably, reducing the dimension will lead to information loss, but keeping too many dimensions will not help mitigate the problems stemming from the curse of dimensionality. As such, it is common in the literature to implement an energy criterion on the spectrum of eigenvalues of the empirical covariance matrix to determine the number of directions to discard. Sometimes known as the Relative Information Content, common thresholds are 99% or 99.9%, as seen in [16]. RIC is defined as the following ratio, with n the number of retained directions in a d-dimensional problem:

$$\text{RIC}(n) = \frac{\sum_{i=1}^{n} \lambda_i}{\sum_{i=1}^{d} \lambda_i} \geq \epsilon \tag{30}$$

Recall that our sample set is drawn from a population with an unknown probability distribution. Therefore, the eigenvalues and eigenvectors are probabilistic. We can then estimate them by eigendecomposition of the empirical covariance matrix $\hat{\mathbf{C}}$. Rigourously, equation (8) should read:

$$\hat{\mathbf{C}} = \frac{1}{N}\sum_{j=1}^{N} \nabla_{\mathbf{x}} f(\mathbf{x}) \nabla_{\mathbf{x}} f(\mathbf{x})^T \tag{31}$$

which can then be decomposed into:

$$\hat{\mathbf{C}} = \hat{\mathbf{W}}\hat{\mathbf{\Lambda}}\hat{\mathbf{W}}^T, \quad \hat{\mathbf{\Lambda}} = \text{diag}(\hat{\lambda}_1, \dots, \hat{\lambda}_d), \quad \hat{\lambda}_1 \geq \cdots \geq \hat{\lambda}_d \geq 0 \tag{32}$$

Since the size of the sampling set is limited, this estimate is biased. One way to estimate this error is to rely on the bootstrap replication technique [17]. The basic idea is to replace the original population with a unique plugin sample. Statistics are obtained using a Monte Carlo technique applied to bootstrap samples from drawing with replacement in the plugin dataset. Applying this procedure to the eigenproblem of equation 31 provides error estimates for the spectral information.

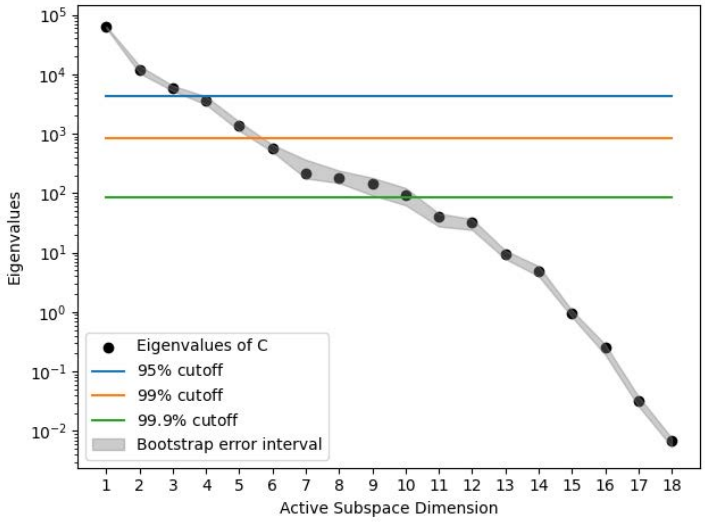


Figure 5: Eigenvalue spectrum and cut-offs based on energy criterion threshold values

These errors are function of the size of the initial dataset, M [10], [18]. Certain considered datasets may be small enough for the bootstrap estimated error on the spectral information to exceed the threshold imposed by the RIC.

We propose to transpose the truncation criterion from spectral information (eigenvalues) to the generalisation error of a response surface trained in the reduced space. The idea is to gradually increase the number of retained dimensions and to monitor the generalisation error. This is done via Q^2, which is analogous to the Stone-Geisser R^2 criterion, but based on model predictions at unseen locations.

$$R^2 = 1 - \frac{\text{Residual Sum of Squares}}{\text{Total Sum of Squares}} = 1 - \frac{\sum_{i=1}^{N}(y[i] - \hat{y}[i])^2}{\sum_{i=1}^{N}(y[i] - \overline{y})^2} \tag{33}$$

$$Q^2 = 1 - \frac{\text{Predicted Residual Sum of Squares}}{\text{Total Sum of Squares}} = 1 - \frac{\sum_{i=1}^{N}(y[i] - \hat{y}_{-i}[i])^2}{\sum_{i=1}^{N}(y[i] - \overline{y})^2} \tag{34}$$

where $\hat{y}$ is the model prediction, $\hat{y}_{-i}$ the prediction by the model where location i has been witheld, and $\overline{y}$ is the mean of all training values y.

Anticipating that even a virtual LOO for Kriging in each subspace size will be expensive, we replace Kriging with cheaper Ridge Regression [19]. The rationale is that both approaches lead to the same truncation, but Ridge Regression will do so cheaply while remaining robust to dataset size. These methods are compared in section 5.3.

5 RESULTS

5.1 Applications

Another problem of high-dimensional problems is difficulty of visualisation. Our first test case is the two dimensional Branin function [7]. While offering limited potential for dimension reduction, it is a well known test case for optimisation, and will help demonstrate certain concepts concerning clustering and overlapping. Being ultimately interested in aerodynamic shape optimisation, we apply the Clustered Active Subspaces method to a standard benchmark in aerodynamics, the NACA0012 aerofoil. This dataset in included in the Active Subspaces Python Toolbox.

5.1.1 The Branin function

$$f(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left[\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 1\right] + 5x_1 \tag{35}$$

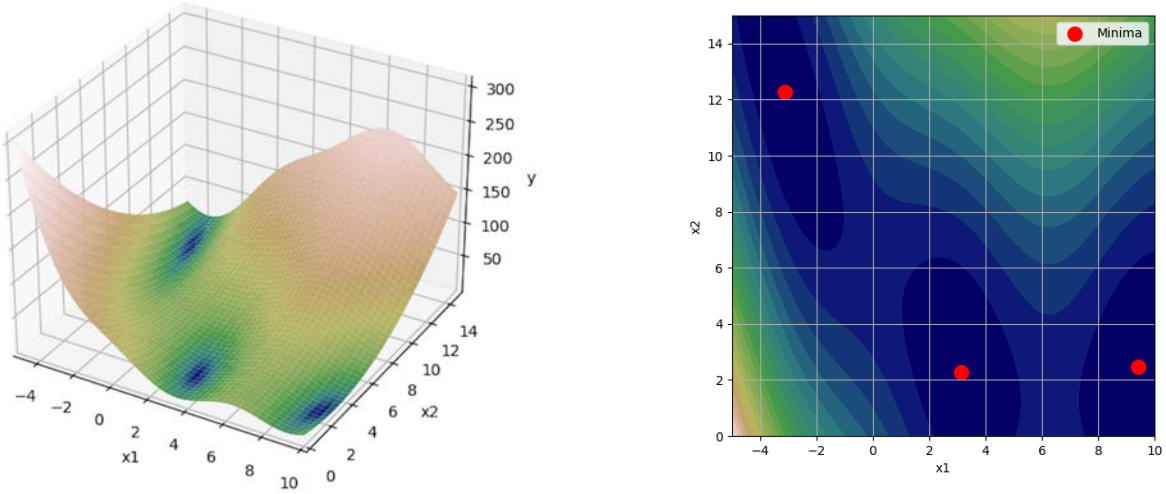


Figure 6: The Branin function

5.1.2 NACA0012 aerofoil

The NACA class of symmetrical aerofoils are well known benchmarks for CFD solvers. This particular implementation of the aerofoil is characterized by 18 Hicks-Henne bumps [3]. The dataset is comprised of 1756 Euler CFD runs, with a space-filling design of experiments. It features two objective functions, lift and drag, and all runs include the computations of the gradients of both objective functions with regards to all control parameters. Gradients are computed using adjoint methods available in SU2 [20]. The lift function exhibits a simple dependence upon the input parameters and is not challenging for dimension reduction. Therefore, in all cases featuring the NACA test case, the function of interest will always be the drag function.

This is a large dataset, and rich considering it is 'only' an 18-dimensional problem. Therefore, we are able to split it into a training set and a testing set. We can thus estimate errors of generalisation without the need for cross validation. The standard train/test split employed is

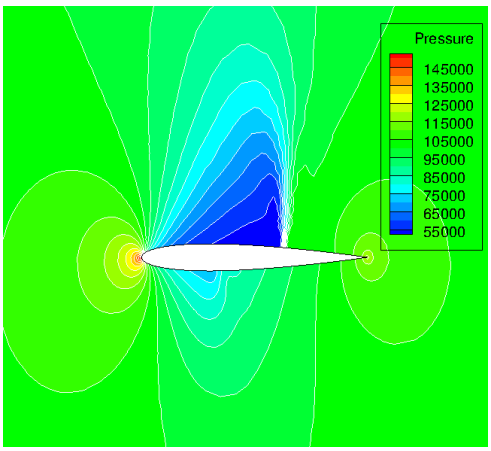


Figure 7: Pressure field around a NACA0012 aerofoil resulting from an inviscid CFD computation with SU2

80-20, which leads to an effective training dataset of 1404 runs. This split is kept the same to ensure repeatability.

5.1.3 ONERA M6 transonic wing

The second physics-based dataset is also included in the Active Subspaces toolbox. It is a classic test for three-dimensional transonic flow, with the added bonus of multiple wind-tunnel tests to which researchers can compare their results. In this particular implementation, the wing is defined by 50 FFD control parameters. The dataset once again features two objective functions, the lift and drag coefficients, as well as their gradients with respect to all 50 design parameters. Due to the increased costs of 3D CFD simulations, this dataset features 300 points.

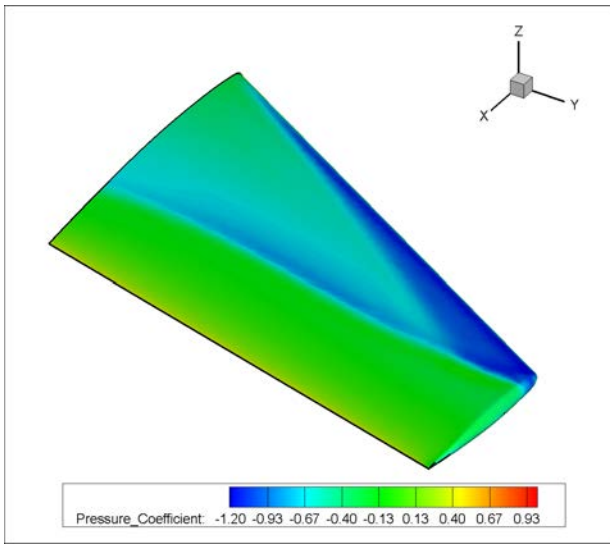


Figure 8: Surface flow on an M6 wing

5.1.4 Data analysis

Before launching oneself into anything, it is vital to approach the data heads-up. This is neither pre-processing since the variables are not altered, nor data mining since we are not dealing with large swaths of data. Yet, this step cannot be ignored. Obviously, a major concern is the potential for dimension reduction in the considered data.

We start by evaluating whether we can realistically expect dimension reduction to succeed in this particular case. We present two such methods.

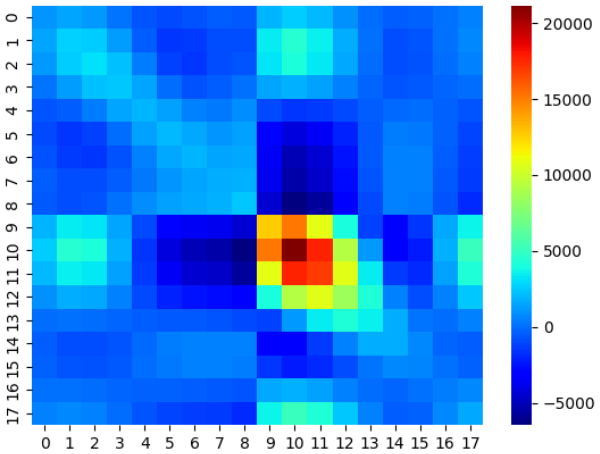


Figure 9: Heatmap of the empirical covariance matrix $\hat{\mathbf{C}}$

Pre-processing's main objective, other than to condition the data set as best as possible in order to limit numerical problems, is feature selection. Well known in data science, feature selection consists in qualitative analysis of the input parameters, as a first step before any exploitation. In our case, it serves a dual purpose: quickly determine the parameters with the most influence on our objective function, and get appreciate the magnitude of dimension reduction for the considered dataset. Methods like the correlation matrix heatmap (figure 9) clearly display the strength of correlation between certain design parameters, which usually indicates some linear combination of these parameters will produce a strongly active component, resulting not only in successful dimension reduction, but also a relevant direction with regards to the objective function.

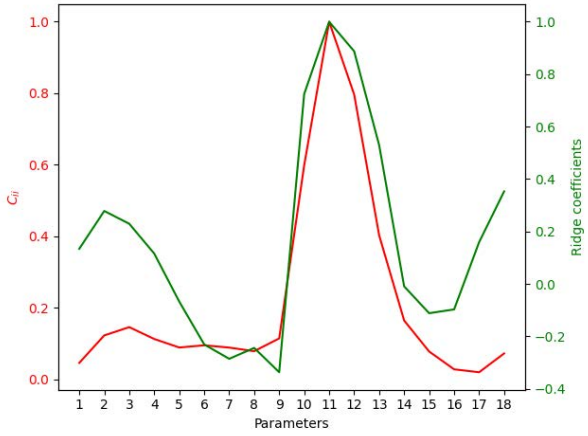


Figure 10: Coefficients of a Ridge regression determined through cross validation (green), and diagonal of the covariance matrix (red)

These results are confirmed by performing a Ridge regression [19] on the data. This is in essence a linear regression with regularisation, whose coefficients give a direct indication of the prominence of each design parameter. We see in figure 10 that parameters 7 through 12 seem to be the most influent. For reference, we have added the scaled values of the diagonal of the empirical covariance matrix $\hat{\mathbf{C}}$, which provides very similar results. Because some parameters seem to be more influent than others, we can expect dimension reduction to succeed.

5.1.5 Classifier accuracy

Increasing the training sets increases accuracy by reducing the the need for the response surfaces to extrapolate. This is beneficial to any recombination method. However, since we are using a smooth recombination which is a linear combination of the neighbouring experts, the weight attributed to each expert is instrumental to the accuracy of the recombined surrogate model. Since we have shown in section 2.3 that we do not use exactly the same functions for the clustering and the recombination, it is legitimate to wonder if some of the recombination error isn't due to this estimation. Since the marginal PoM classifier is based on the same principle as the joint PoM classifier ($j^* = \arg\max_{j\in[1,K]} \text{PoM}_j$), our reasoning is that misclassified points indicate inaccuracies in the β_k functions for these particular locations. Identifying misclassified points can help accuracy, since these functions are used to weigh the contributions of each local expert to the global surrogate model (see eq.23). A useful tool to evualuate the accuracy of a classifier is the confusion matrix. A confusion matrix compares the percentage of correctly classified points from each cluster. A perfect classifier has a diagonal matrix: for each cluster, 100% of the points have been correctly classified. In all cases considered (Branin, NACA, M6 ing), the matrix is very diagonal for a host of cluster numbers; the β_ks are deemed pertinent.

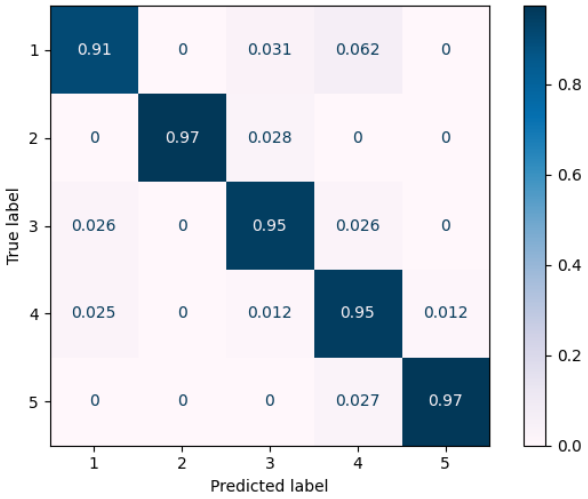


Figure 11: Confusion matrix of the marginal probability of membership classifier, in the 5 cluster case

We now contrast this with the confusion matrix between of the support vector classifier. We repeat the procedure with two widely used kernel options, a linear separator and an RBF-based kernel. The blue outlines are the probable locations of the cluster boundaries, as computed by the Gaussian Mixture model, while the shaded contours determine the boundaries as computed by SVM.

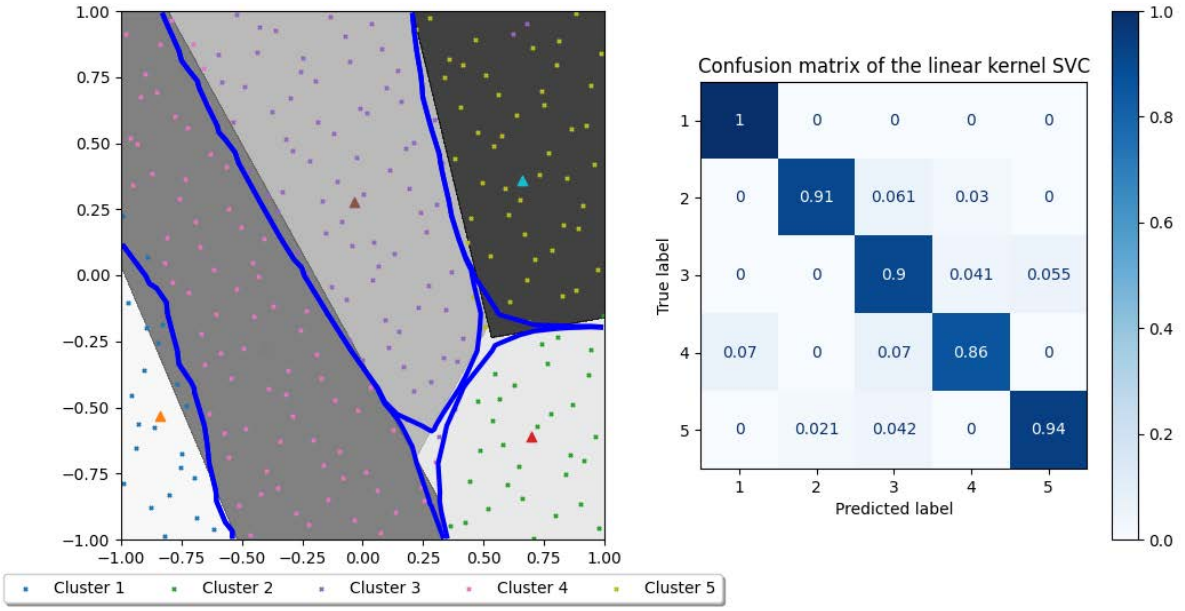


Figure 12: Comparing the cluster boundaries calculated by the marginal probability of membership (blue lines) with boundaries obtained by the linear kernel support vector classifier. Also featured is the confusion matrix of the linear kernel support vector classifier in the 5 cluster case

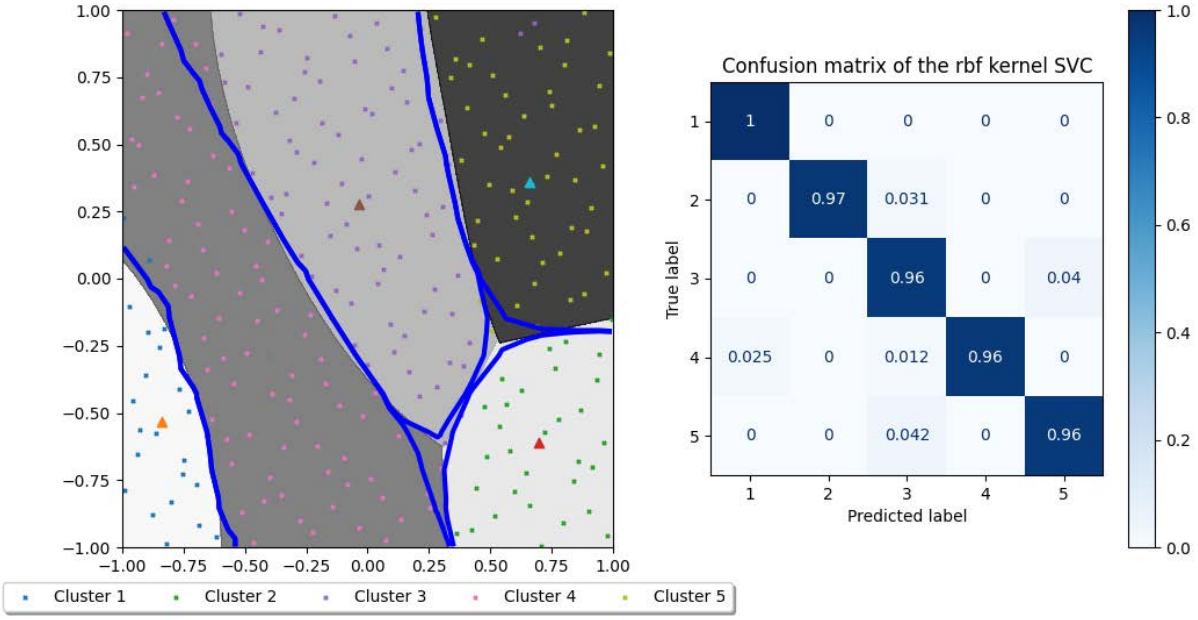


Figure 13: Comparing the cluster boundaries calculated by the marginal probability of membership (blue lines) with boundaries obtained by the RBF kernel support vector classifier. Also featured is the confusion matrix of the RBF kernel support vector classifier in the 5 cluster case

Since our dataset is not linearly separable, the RBF kernel allows for more accurate classification than its linear counterpart. The confusion matrix is an indication of the learning error of classifiers. While this is an important metric, we are more interested by the generalisation error: how do the classifiers perform on points that have not been used to train them?

To this end, we generate a wider training set, perform an 80-20 train/test split, and ask the classifiers to predict the label of unseen points. We then compare these predictions to their actual classification as computed by the joint law of membership. Table 1 concerns the 2D Branin test case, and table 2 focuses on the NACA dataset.

Number of clusters	Method	Learning error	Generalisation error
4 clusters	Marginal PoM	8.00 %	12.28 %
	Linear SVC	10.22 %	12.28 %
	RBF SVC	2.22 %	8.77 %
5 clusters	Marginal PoM	5.33 %	8.77 %
	Linear SVC	10.22 %	14.04 %
	RBF SVC	2.22 %	12.28 %
10 clusters	Marginal PoM	4.44 %	10.52 %
	Linear SVC	14.22 %	26.32 %
	RBF SVC	7.99 %	15.78 %
20 clusters	Marginal PoM	3.11 %	28.07 %
	Linear SVC	30.66 %	38.59 %
	RBF SVC	7.99 %	22.81 %

Table 1: Comparing classifier accuracy on the Branin dataset, 80-20 split

Number of clusters	Method	Learning error	Generalisation error
3 clusters	Marginal PoM	9.76 %	14.77 %
	Linear SVC	26.86 %	33.52 %
	RBF SVC	4.20 %	17.89 %
5 clusters	Marginal PoM	9.47 %	22.15 %
	Linear SVC	31.27 %	40.06 %
	RBF SVC	4.84 %	28.69 %
10 clusters	Marginal PoM	9.47 %	18.18 %
	Linear SVC	25.28 %	40.06 %
	RBF SVC	4.42 %	34.66 %
20 clusters	Marginal PoM	2.06 %	23.58 %
	Linear SVC	9.18 %	49.72 %
	RBF SVC	3.06 %	46.6 %

Table 2: Comparing classifier accuracy on the NACA dataset, 80-20 split

This confirms that the marginal probability of membership classifier, $j^* = \arg\max_{j\in[1,K]} \beta_j$ is robust to different cases.

Nonetheless, we propose two areas of development.

β_k switching for misclassified points Firstly, points are misclassified by the marginal PoM classifier because of inaccuracies of the marginal probabilities of membership, β_k. This is problematic since these $\beta_k(\mathbf{x})$ are used to weigh the Kriging models, so these errors will be propagated in the reconstruction. We correct the weighting attributed to each expert by attributing the highest marginal probability of membership to the location's actual cluster. This changes only the values of the two β_ks, for the sites which have been misclassified. While this little trick cannot help predictive performance of the model at new locations, it should improve the learning error, leading to a more accurate surrogate model.

Weighted SVC The SVC formalism enables a extension to reduce the number of misclassifications. The SVC allows for a weighting of the locations, to ensure certain samples deemed more important are correctly classified. The weighting comes from the true (joint law of (X, Y)) probability of membership:

$$p_{i,j} = \frac{\pi\Big((X,Y) = (x_i, y_i) \mid \kappa = j\Big)}{\sum_{k=1}^{K} \pi\Big((X,Y) = (x_i, y_i) \mid \kappa = k\Big)} \tag{36}$$

We can also use the probability of generating a point, knowing that it belongs to the current cluster. The weighting would then be:

$$p_i = \pi(X = x_i \mid \kappa = k), \qquad \pi(X = x \mid \kappa = k) = \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{37}$$

In doing so, we are in essence trying to force the correct classification of points with the lowest probabilities of membership. In all rigor, the weighting we use is $1 - p_{i,j}$, since were are shifting importance towards uncertain points. This extension deserves more time to be correctly analysed, and will therefore be presented in a future publication.

5.2 Accuracy gains through probability of membership based overlapping

Consider first the NACA dataset, from a purely clustering point of view. At 1404 points for 18 dimensions, it is still tractable for standard responses such as Kriging. As such, a single global Kriging model over the entire design space will serve as the benchmark (blue curves). To this benchmark we compare both recombination strategies (hard, in red, and smooth, in green) implemented in the SMT surrogate modelling toolbox [21]. Finally, the orange curves correspond to the overlapped Kriging method detailed in section 3. We successively apply these three methods for an increasing number of clusters, and use the generalisation error

$$\text{RMSE} = \frac{1}{N_{\text{test}}} \sqrt{\sum_{i=1}^{N_{\text{test}}} \left(y_i - \hat{y}_i\right)^2} \tag{38}$$

as our performance metric, with y_i and $\hat{y}_i$ respectively denoting the true function value and the model's prediction at $\mathbf{x}_i$.

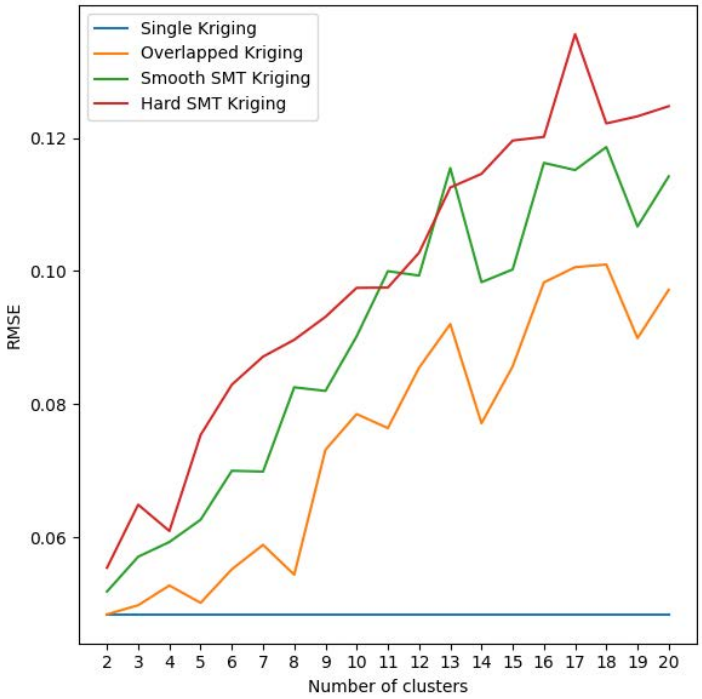


Figure 14: Generalisation error of standard and improved methods for local expert recombination. The orange curve shows the benefit of overlapping over standard recombination methods.

Drag coefficient is not expected to feature any discontinuities, which is why the hard recombination presents the most error. Smooth recombination improves upon these performances, but Kriging struggles with extrapolation, and boundaries between clusters are mis-modelled. For the overlapping, we chose the cardinal criterion, which requires the addition of 50% of points to each cluster. As hoped, overlapping is beneficial to accuracy, while not impacting training time all that much. Even more impressively, for 4 and 5 clusters, overlapped Kriging is as accurate as the single global Kriging model.

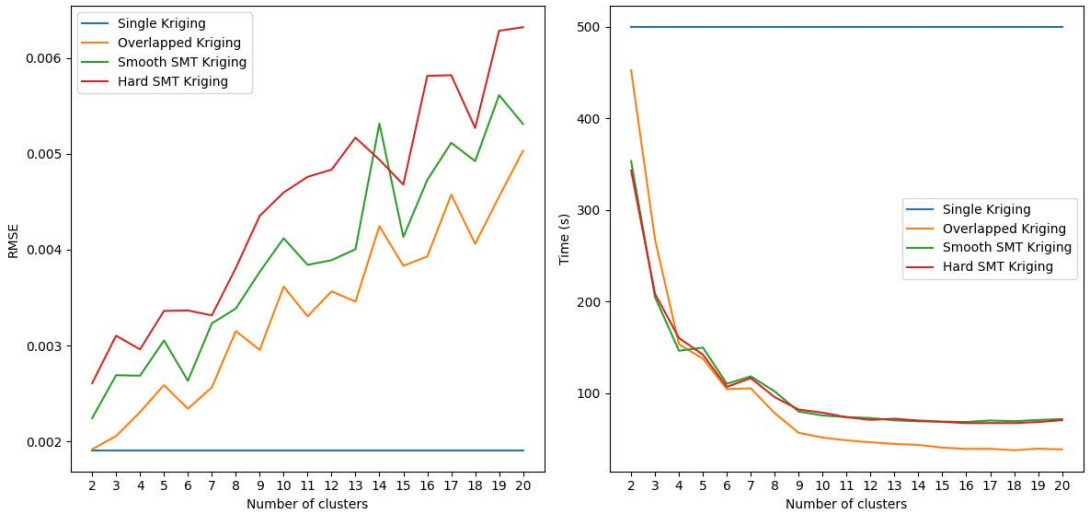


Figure 15: Generalisation error and computation time of standard and improved methods for local expert recombination, using PoM threshold criterion

The cardinal criterion can be problematic since it forces the addition of points, regardless of whether the new points are actually beneficial. To circumvent this, we implement a probability threshold criterion. All points more likely than a certain threshold γ to belong to the current cluster are added to the local response surface's training set. This is more flexible: if no pertinent points are found, none are added. If a lot of points could benefit the response surface, they are all added.

Similarly to when using the cardinal criterion, using the probability threshold criterion leads to improved accuracy over the other clustered methods. However, while accuracy is good, it does not intersect the blue curve. Judging by the computation time, we believe this is because few points are actually added to the training sets, since very few points are likely enough to belong to other clusters than their own. Excessively lowering the threshold will lead to increased computation times, and adding points which may not be beneficial to accuracy. This leads to an important set of questions: which criterion to choose, and what value needs to be set?

5.3 Active Subspace dimension selection criterion

In the absence of a dimension selection criterion, the only option is to train a Kriging model in every possible subspace size, then compare accuracy and computation time to determine the optimal subspace size. Figure 16 below shows the generalisation error of the Kriging models as a function of the size of the active subspace for drag coefficient around the NACA aerofoil. The training was done on 80% of the sample, which yields a training set of 1404 points and a test set of 352 points.

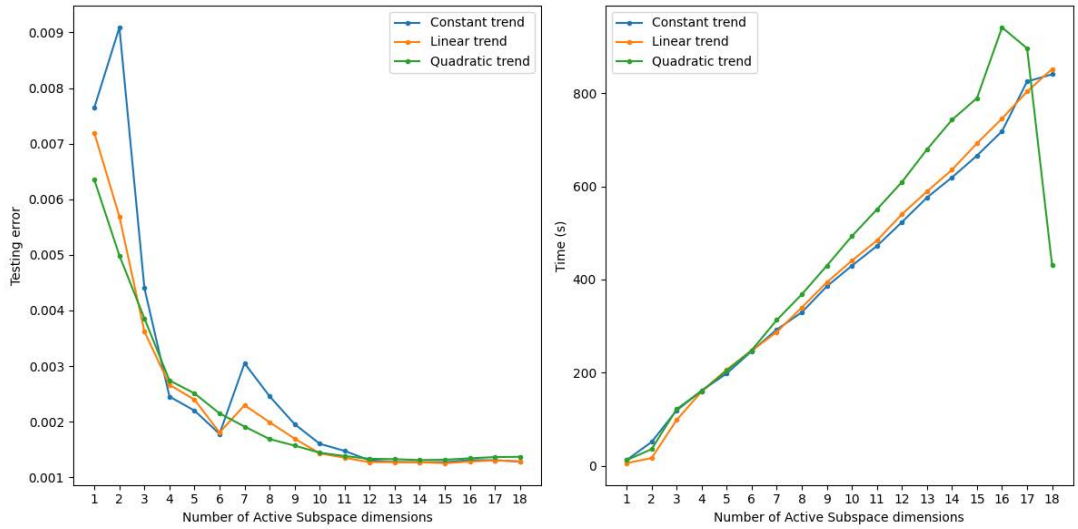


Figure 16: Generalisation error and computation time for Kriging for different active subspace sizes. Predictably, larger active subspaces lead to more accurate Kriging models, no matter the trend type.

This is the benchmark to which we will compare the criteria presented in section 4. Three common thresholds for the energy criterion are also included in the comparison.

Method	Suggested Dimension	Criterion time	Generalisation error	Total time
95% cut-off	4	$\emptyset$	2.45×10^{-3}	161s
99% cut-off	6	$\emptyset$	1.78×10^{-3}	246s
99.9% cut-off	11	$\emptyset$	1.47×10^{-3}	472s
Single Ridge	3	10^{-2}s	4.4×10^{-3}	119s
LOO-Ridge	4	17s	2.45×10^{-3}	178s
RBF	4	56s	2.45×10^{-3}	217s
Virtual LOO	4	400s	2.45×10^{-3}	561s
Kriging	6	787s	1.78×10^{-3}	787s

Table 3: Comparing dimension selection criteria on the global 18-dimensional NACA dataset

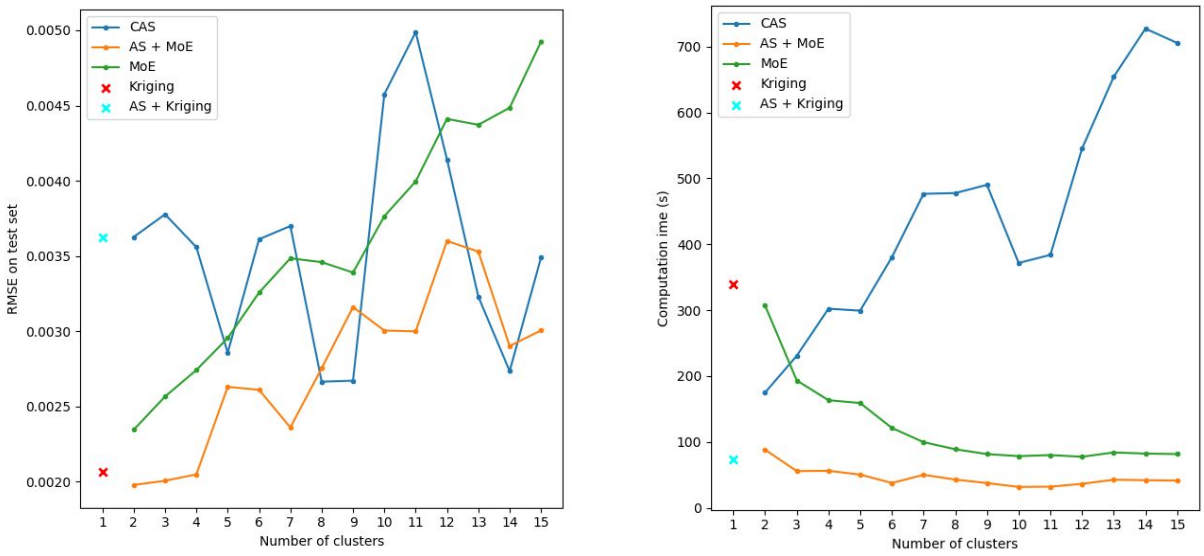
The regression criterions (Ridge and RBF) are capable of identifying pertinent active subspace sizes affordably, confirming our idea has merit. In the presented case, the dataset is very large, which means the discovery of the eigenvalues is very precise (see bootstrap interval in figure 5), enabling the energy criterion to perform well, but this may not always be the case.

5.4 Putting it all together: the clustered active subspaces method

We now put to use the developments presented in the two previous subsections. Decision on the number of clusters for now still requires user input, so we iterate over the number of clusters. For each run, after the number of clusters has been set, the local active subspaces are discovered and the clustered dataset is projected into these reduced spaces. Pertinent nearby points are then added to the training sets using the overlapping procedure detailed in section

3. The Kriging models are trained, then recombined. The train/test split is the same as in section 5.2. To our clustered active subspaces method, we compare the performance of using a single global active subspace and corresponding surrogate model. Performance metrics are the learning error (RMSE on the training set), generalisation error (RMSE on the unseen test set) and computation time.

We first present the behaviour of the different surrogate model methods as a function of the number of clusters used to partition the design space. We compare the CAS method (in blue) with a single Kriging model trained on the entire design space (red cross), a Kriging model trained in a single global active subspace (cyan cross), a mixture of Kriging models trained in the initial space (green), and a mixture of experts trained on the local active subspaces (orange). The benchmarks are the single Kriging model and the global AS + Kriging, both corresponding to the one cluster case. The CAS method features clustering, dimension reduction, and overlapping. No other method considered here features all three.



(a) Generalisation error

(b) Computation time

Figure 17: Generalisation error and computation time of the surrogate models on the NACA case.

The NACA case features a very large training set, for a reasonable amount of directions. A single Kriging model performs well, and is the reference in terms of accuracy. Clustering and the use of local active subspaces is, in most cases, beneficial to accuracy when compared to the global Active Subspace + Kriging method. However, in the overlapped case, one must be careful when deciding which points to add, as adding the wrong points can be detrimental to predictive power. These results are consolidated in the following table.

Method	Learning Error	Generalisation error	Computation time
Single Kriging	9.00×10^{-15}	2.06×10^{-3}	339s
MoE	9.60×10^{-4}	2.34×10^{-3}	252s
AS + Kriging	3.00×10^{-15}	3.62×10^{-3}	73s
Clustered Active Subspaces	6.80×10^{-4}	3.72×10^{-3}	356s
Clustered AS + MoE	1.44×10^{-3}	1.82×10^{-3}	71s

Table 4: Comparing surrogate models for the drag coefficient on the 18-dimensional, 1400 point NACA test case

ONERA M6 wing Taking these methods to the 50 dimensional ONERA M6 wing presents some challenges. Firstly, the 50 dimensions should be more of a challenge for a single classic surrogate model such as Kriging. The much lower number of points in the dataset is also a good challenge.

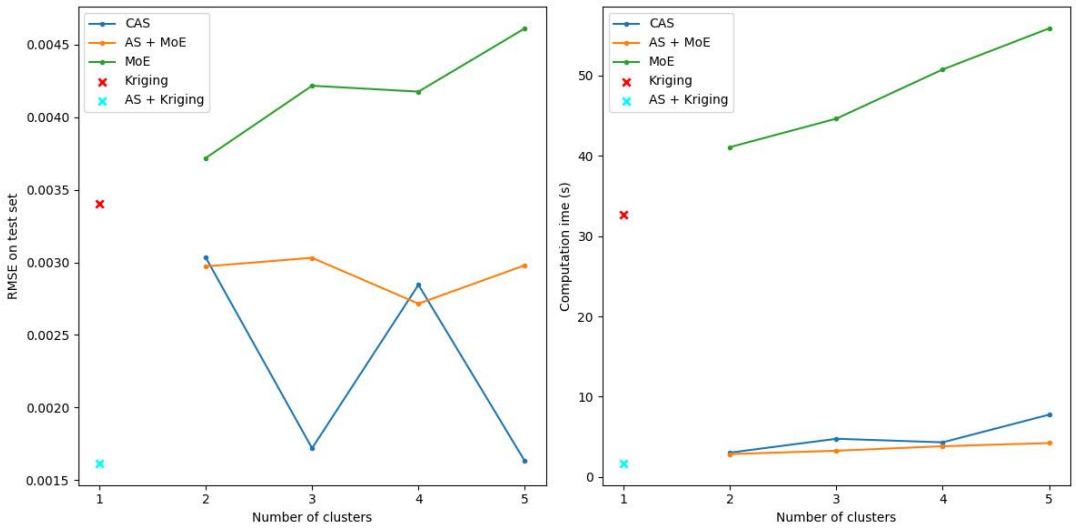


Figure 18: Generalisation error and computation time of the different surrogate models on the M6 case

Figures 18 and 19 compare the CAS method with the same surrogate models as in the NACA test case above, but this time compares their learning error as well as their generalisation error.

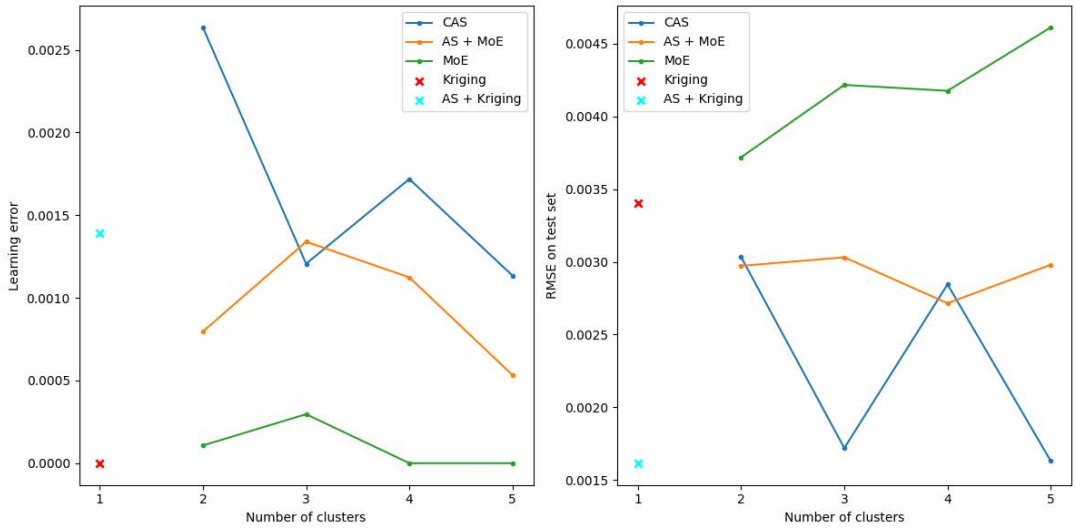


Figure 19: Learning and generalisation error of the different surrogate models on the M6 case

Three and five clusters offer the best predictive power for the CAS method, at reasonable cost. Note how as opposed to the NACA case, the Kriging model has less predictive power in the original domain than in the reduced space, due to the increase in dimension. The learning error is larger for the methods which include dimension reduction. This is because when reducing dimension, regressive Kriging must be used, whereas training the Kriging model in the original domain enables the use of interpolation. Exact results are presented in the table below.

Method	Learning Error	Generalisation error	Computation time
Single Kriging	6.0×10^{-16}	3.4×10^{-3}	32.7s
MoE	1.0×10^{-4}	3.7×10^{-3}	41.1s
AS + Kriging	1.4×10^{-3}	1.6×10^{-3}	1.71s
Clustered AS + MoE	4.0×10^{-16}	4.2×10^{-3}	3.84s
Clustered Active Subspaces	1.1×10^{-3}	1.6×10^{-3}	7.76s

Table 5: Comparing surrogate models for the drag coefficient on the 50-dimensional, 270 point M6 test case

6 CONCLUSIONS

This paper presents a novel way to build a global surrogate model in high dimension. Clustering the design space using the joint probability of inputs and outputs is beneficial on several fronts. Firstly, it helps deal with multimodality by making decisions based on function topology, even in cases where the design of experiments is non informative (space filling DoEs, for example). Second, clustering benefits the Active Subspaces dimension reduction method by applying it locally, where gradient information is most pertinent.

However, we found that clustering can lead to prediction errors at the boundaries between clusters, where the local experts are asked to extrapolate. Therefore, one of the main objectives of this work was to manage the inaccuracies borne of recombination. This was achieved by extending the training sets of the local experts, leading to patches around cluster boundaries

where two local experts overlap. This overlapping was shown to be beneficial to predictive power, and, more importantly for us, we have shown that Clustered Active Subspaces can improve upon existing methods of dimension reduction for surrogate modelling.

In particular, the CAS method exhibits improved predictive power for two benchmark aerodynamic test cases, while not impacting the computation time. We have built an accurate surrogate model over a large design space, which we postulate is by construction more robust to multimodality due to its clustered nature.

7 PERSPECTIVES

The work has highlighted new development perspectives. First of all, the Gaussian Mixture Model has so far has been built on the joint law of inputs and outputs, $\pi(\mathbf{x}, f(\mathbf{x}))$, but we could have considered the joint law of inputs and the output's gradients ($\pi(\mathbf{x}, \nabla f)$). We plan to compare these two options, as this could lead to different cluster boundaries, and perhaps improved accuracy. The latter has the added benefit of being coherent with our use of Active Subspaces as a dimension reduction strategy.

Next, keeping with the trend of maximising the use of gradient information, we will test out the performance gains to be had by using gradient information to build our response surface, through the use of Gradient Enhanced Kriging.

Overlapping through SVM showed promise, but requires further work to fully exploit the decision function and determine the size of the overlapping patches.

References

[1] A. Dumont, J.-L. Hantrais-Gervois, P.-Y. Passaggia, J. Peter, I. Salah el Din, and É. Savin, "Ordinary kriging surrogates in aerodynamics," in *Uncertainty Management for Robust Industrial Design in Aeronautics : Findings and Best Practice Collected During UMRIDA, a Collaborative Research Project (2013–2016) Funded by the European Union*, C. Hirsch, D. Wunsch, J. Szumbarski, Ł. Łaniewski-Wołłk, and J. Pons-Prats, Eds. Cham: Springer International Publishing, 2019, pp. 229–245, ISBN: 978-3-319-77767-2. DOI: 10.1007/978-3-319-77767-2_14. [Online]. Available: https://doi.org/10.1007/978-3-319-77767-2_14.

[2] T. Lukaczyk, F. Palacios, J. Alonso, and P. Constantine, "Active subspaces for shape optimization," Jan. 2014. DOI: 10.2514/6.2014-1171.

[3] P. Constantine, "Active subspace methods in theory and practice: Applications to kriging surfaces," *SIAM Journal on Scientific Computing*, 2014. DOI: 10.1137/130916138.

[4] H. Wold, "Estimation of principal components and related models by iterative least squares," *Multivariate Analysis*, ed. by P. R. Krishnajah, pp. 391–420, 1966.

[5] L. Cambier, S. Heib, and S. Plot, "The onera elsa cfd software: Input from research and feedback from industry," *Mechanics and Industry*, vol. 14, no. 3, pp. 159–174, 2013. DOI: 10.1051/meca/2013056.

[6] M. Jadoui, C. Blondeau, E. Martin, and F.-X. Roux, "Comparative study of inner-outer krylov solvers for linear systems in structured and high-order unstructured cfd problems," *Computers and Fluids*, vol. 244, 2022. DOI: 10.1016/j.compfluid.2022.105575.

[7] A. Forrester, A. Sobester, and A. Keane, *Engineering design via surrogate modelling: a practice guide*, Wiley, Ed. 2008.

[8] *Onera m6 wing, \NoCaseChange{https://www.onera.fr/fr/actualites/laile-onera-m6-star-de-la-cfd}*, Accessed: 2023-02-08.

[9] C.E.Rasmussen and K. I. Williams, *Gaussian Processes for Machine Learning*. 2006.

[10] P. G. Constantine, *Active Subspaces: Emerging ideas for Dimension Reduction in Parameter Studies, Emerging ideas for Dimension Reduction in Parameter Studies*. SIAM Spotlights, 2015.

[11] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 39, no. 1, pp. 1–22, 1977. DOI: https://doi.org/10.1111/j.2517-6161.1977.tb01600.x. eprint: https://rss.onlinelibrary.wiley.com/doi/pdf/10.1111/j.2517-6161.1977.tb01600.x. [Online]. Available: https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/j.2517-6161.1977.tb01600.x.

[12] M. Jordan and R. Jacobs, "Hierarchical mixtures of expert and the em algorithm," *Neural Computation*, vol. 6, Aug. 2001. DOI: 10.1162/neco.1994.6.2.181.

[13] S. Kotsiantis, I. Zaharakis, and P. Pintelas, "Machine learning: A review of classification and combining techniques," *Artificial Intelligence Review*, vol. 26, pp. 159–190, Nov. 2006. DOI: 10.1007/s10462-007-9052-3.

[14] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the fifth annual workshop on Computational learning theory*, 1992, pp. 144–152.

[15] D. Bettebghor and F.-H. Leroy, "Overlapping radial basis function interpolants for spectrally accurate approximation of functions of eigenvalues with application to buckling of composite plates," *Computers & Mathematics with Applications*, vol. 67, no. 10, pp. 1816–1836, 2014, ISSN: 0898-1221. DOI: https://doi.org/10.1016/j.camwa.2014.03.020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0898122114001539.

[16] M. Bergmann, A. Ferrero, A. Iollo, E. Lombardi, A. Scardigli, and H. Telib, "A zonal galerkin-free pod model for incompressible flows," *Journal of Computational Physics*, vol. 352, pp. 301–325, 2018, ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2017.10.001. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0021999117307386.

[17] B. Efron and R. Tibshirani, *An Introduction to the Bootstrap*. CRC Press, 1994.

[18] G. H. Golub and C. F. Van Loan, *Matrix Computation*. 1996, pp. 397–399.

[19] A. E. Hoerl and R. W. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, vol. 12, pp. 55–67, 1970.

[20] F. Palacios, M. Colonno, A. Aranake, *et al.*, "Stanford university unstructured (su2): An open-source integrated computational environment for multi-physics simulation and design," Jan. 2013. DOI: 10.2514/6.2013-287.

[21] M. A. Bouhlel, J. T. Hwang, N. Bartoli, R. Lafage, J. Morlier, and J. R. R. A. Martins, "A python surrogate modeling framework with derivatives," *Advances in Engineering Software*, p. 102 662, 2019, ISSN: 0965-9978. DOI: https://doi.org/10.1016/j.advengsoft.2019.03.005.