AIAA AVIATION Forum
13-17 June 2016, Washington, D.C.
17th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference

10.2514/6.2016-4001

Improvement of efficient global optimization with mixture of experts: methodology developments and preliminary results in aircraft wing design

N. Bartoli [*], I. Kurek[†], R. Lafage[*], T. Lefebvre[*], R. Priem[†]
Onera, The French Aerospace Lab, Toulouse, France

M.-A. Bouhlel[‡]
University of Michigan, Ann Arbor, MI, USA

J. Morlier[§], V. Stilz[¶]
Université de Toulouse, Institut Clément Ader (ICA), CNRS, ISAE-SUPAERO, Toulouse, France

R. Regis [‖]
Saint Joseph's University, Philadelphia, Pennsylvania 19131, USA

For decades, numerical tool improvements enabled the optimization of complex processes occurring during the conceptual phase. Nowadays simulators can determine numerous coupled physical effects with high accuracy and allow cheap and fast virtual testing. However, high fidelity tools require long computation times (several days of computation using High Performance Computing solutions) and thus optimization based on these high fidelity tools is often done at higher computational cost (gradient based). This work aims at optimizing a complex design using costly simulation codes given a fixed computational budget. In aeronautical engineering these codes can be coupled in space (such as Fluid Structure Interaction) and/or in time (for transient analysis). The fixed budget implies the use of surrogate-based method with adaptive sampling in order to promote a trade-off between exploration and exploitation. The proposed optimization is based on a sequential enrichment approach (typically Efficient Global Optimization), using an adaptive mixture of kriging-based models. The strategy relies on an improvement of the kriging model that enables the handling of a large number of design variables whilst maintaining rapidity and accuracy. A key feature is the use of mixture of experts technique to combine local surrogate models to approximate both the objective function and the constraints. Our strategy will be introduced through mathematical methods and detailed algorithms presentation. Finally, we produce several validations on analytical test cases (supervised) and two extensions such as the well-known MOPTA test case from automotive industry and aircraft wing structural optimization. The experiments confirm that the proposed global optimization approach minimizes the number of black box evaluations and in this sense it is well suited for high-dimensional problems with a large number of constraints.

I. Symbols and notations

Matrices and vectors are in bold type.

[*]Research Engineer, System Design and Performance evaluation Department, AIAA Member.
[†]Student, System Design and Performance evaluation Department
[‡]Postdoctoral Fellow, Department of Aerospace Engineering.
[§]Professor, Structural Mechanics and AIAA Member.
[¶]Student, ISAE
[‖]Associate Professor of Mathematics

Symbol	Meaning
n	Number of sampling points
d	Dimension
$\mathbf{x}$	$1 \times d$ vector
x_j	j^{th} element of a vector $\mathbf{x}$
$\mathbf{X}$	$n \times d$ matrix containing sampling points
$\mathbf{y}$	$n \times 1$ vector containing simulation of $\mathbf{X}$
$\mathbf{x}^{(i)}$	i^{th} training point for $i = 1, \ldots, n$ (a $1 \times d$ vector)
f	Objective function of the optimization problem
m	Number of constraints in the optimization problem
c_i	i^{th} constraint function for $i = 1, \ldots, m$
$k(.,.)$	Covariance function
$\mathcal{N}(0, k(.,.))$	Distribution of a Gaussian process with mean function 0 and covariance function $k(.,.)$
$\mathbf{x}^t$	Superscript t denotes the transpose operation of the vector $\mathbf{x}$
x^*	Superscript $\star$ denotes the optimal value of the variable x
$\hat{y}$	Superscript $\hat{}$ denotes the approximation (regression or interpolation models) of the output y

II. Introduction

In the last decade, Onera has initiated an important methodological effort to improve the efficiency of vehicle design processes through the development of tools and techniques in the field of multidisciplinary optimization.[1] In the frame of new aircraft configurations development (such as strut-braced wing aircraft), the integration of more accurate data coming from high fidelity analysis earlier in the design process seems to be compulsory.[2] Indeed, some important modifications at the configuration level take place and new methods have been investigated to optimize coupled disciplines on aircraft design as multidisciplinary and multi-fidelity optimization. High fidelity codes such as Finite Element Method (FEM) and Computational Fluid Dynamics (CFD) codes are used earlier in the process. Therefore, solutions have to be investigated to optimize such complex systems (such as aero-structural wing design) at limited budget but using possibly several costly simulation codes.

This paper proposes a novel optimization method that is based on a sequential enrichment approach (typically based on the Efficient Global Optimization (EGO) method[3]), using an adaptive surrogate model in order to control the exploration/exploitation infill criteria.[4] The strategy relies on an improvement of the kriging model that enables the handling of a large number of design variables whilst maintaining rapidity and accuracy. A key feature is the use of a mixture of experts technique to combine local surrogate models to approximate both the objective function and the constraints. The proposed optimization method is called SEGOMOE and it combines the approach used by the SuperEGO[5] (SEGO) algorithm with the Mixture of Experts[6] (MOE) approach based on multiple surrogate models.

First, the optimization process is presented with a focus on the improvements made on both the surrogate models and the EGO approach. Surrogate models, such as kriging,[7,8] are widely used in engineering problems[9] to substitute time-consuming high fidelity models. To handle large number of design variables involved in multidisciplinary optimization using different levels of fidelity, an evolution of the universal kriging is used. Tuning the kriging hyper-parameters involved in the correlation function by maximum likelihood (or cross validation), can be time consuming especially when the dimension increases. A recent technique, called KPLS that accelerates the calculation, consists of a combination of the Partial Least Squares (PLS) method and the kriging model.[10] Partial Least Square regression (PLSR) is a well-known tool[11] for high dimensional problems by projecting the design variables in a space of smaller dimension whilst investigating correlation between input and output variables. In this way, the PLSR gives information on any variable contribution by exploiting the linear relationship between input and output variables. The main idea developed in 10 and 12 consists of combining this information with the hyper-parameters of kriging, the resulting approaches are thus called KPLS and KPLS-K. Hence, the kriging spatial correlation function is replaced by a new correlation function within variables that have a reduced dimension. Latent variables transcribe influence of initial variables to output ones and trim the dimension to h, the number of latent variables with h up to 4

and the computation time is thus reduced. Based on kriging properties to predict both the output variable and an estimate of variance, the Efficient Global Optimization algorithm (so called EGO and described by Jones et al.[3]) relies on an Expected Improvement (EI) criterion accounting for the exploration-exploitation trade-off. One major problem with using the EGO algorithm is that it cannot handle constraints in its standard version while the SuperEGO algorithm and our proposed approach are designed for problems with inequality constraints. The main features of our proposed SEGOMOE algorithm are the following:

1. The kriging model is replaced by some proposed models (KPLS and KPLS-K) to approximate both the objective function and all the constraints.

2. A combination is also proposed in order to increase the accuracy of the approximation by replacing a single global model by a weighted sum of local models. This technique known as mixture of experts is based on a partition of the problem domain into several subdomains via clustering algorithms, and this is followed by local expert training in each subdomain.

3. Different criteria are used for selecting infill sample points like the Watson and Barnes criterion (WB2, see 5) to give slightly more merit to local search.

4. The search of the optimum is done using a derivative free optimizer (such as COBYLA for Constrained Optimization BY Linear Approximation, see 13) capable of considering non linear constraints.

Finally, the proposed SEGOMOE algorithm is assessed on various optimization problems, ranging from analytical tests cases in high dimensions to the well-known MOPTA test case from automotive industry[14] and an aircraft wing design optimization. For solving problems with a large number of design variables, the recent approach developed in 15 consists of coupling an iterative optimization method (Mid-Range Approximation Method) that makes use of trust-regions with an efficient way of computing Gradient Enhanced Kriging (Compound Search Method). The results presented are promising in terms of number of design variables (100+) but it requires the knowledge of both the function values and their derivatives (from adjoint method for instance) at each sampling point. This assumption is not necessary in the method presented here.

The paper begins with a recall on the surrogate based optimization techniques with description of the main steps from the initial design of experiments to the handling of constraints. Next, we discuss the approach taken to implement and solve high dimensional problems with a large number of constraints by listing all the improvements, as well as the measures taken to address them. Finally, we present some optimization results that demonstrate the validity of the SEGOMOE approach advocated in this paper and its potential for wing aircraft design.

III. Surrogate based optimization

Our proposed method is intended to solve the following nonlinear optimization problem, which represents a physical optimization problem and is formulated as follows:

$$\begin{cases} \min\limits_{\mathbf{x}\in\mathbb{R}^d} f(\mathbf{x}) \\ s.t. \\ c_1(\mathbf{x}) \leq 0 \\ \vdots \\ c_m(\mathbf{x}) \leq 0 \end{cases} \tag{1}$$

where $f(\mathbf{x})$ is the objective function, which is to minimized, $c_i(\mathbf{x})$ corresponds to the i^{th} constraint ($\forall i \in [1,\ldots,m]$) where m is the number of constraints and $\mathbf{x} \in \mathbb{R}^d$. The objective and the constraint functions could be given by expensive simulators such as high-fidelity aerodynamic or aerostructural models... For instance, in some bi-disciplinary problems, the objective function f could be given by a linear or non-linear combination of the disciplinary outputs such as the structural weight and lift-to-drag ratio.

In order to solve this optimization problem, we first describe how to construct some accurate surrogate models based on a few sampling points well chosen to describe the problem domain. Then, the enrichment procedure to minimize the objective function is described in both cases without and with constraint functions.

A. DOE

Building Design Of Experiments (DOE) is a major step for construction of surrogate models. Moreover, DOE have a significant effects on the accuracy of the surrogate models. In particular, the Latin Hypercube Sampling (LHS) is widely used for engineering problems since they are simple to build. In the last decades, several researchers try to improve the space-filling properties while maintaining a low computational cost.[16–18] In this study, we used the Enhanced Stochastic Evolutionary (ESE) algorithm designed by Jin et al. in 2005[19] for building the LHS. This algorithm improves the LHS quality by optimizing a specific criterion, based on the computation of the distances between points. The ESE algorithm is inspired from the simulated annealing.[20]

B. Kriging models

The kriging model (or the more general Gaussian process[21] regression) is an interpolation method where the interpolated values are modeled by a Gaussian process with mean $\mu(.)$ governed by a prior covariance kernel $k(.,.)$, which depends on some parameters θ to be determined. Under suitable assumptions on the priors, the kriging model gives the best linear unbiased prediction of the intermediate values. The theoretical basis of the method was developed by G. Matheron[7] based on D.G. Krige's work.[22] The method has been then widely used in the domain of computer simulation and machine learning.[8,23,24]

Let the sample data points be given by the matrix $\mathbf{X} = [\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}]^t$, with $\mathbf{x}^{(i)} \in \mathbb{R}^d$ for $i = 1, \ldots, n$, and let the expensive observed responses be given by the vector $\mathbf{y} = [y^{(1)}, \ldots, y^{(n)}]^t$, with $y(\mathbf{x}^{(i)}) = y^{(i)}$ for $i = 1, \ldots, n$. Next, we define the stochastic process $Y(x) = \mu(x) + Z(x)$ with $\mu(\mathbf{x})$ an unknown linear or nonlinear function and $Z(x)$ a realization of a stochastic Gaussian process with $Z \sim \mathcal{N}(0, \sigma^2)$. In this study, we consider only the ordinary kriging model, which is a particular case, where $\mu(\mathbf{x}) = \mu$ is an unknown constant, $\forall \mathbf{x} \in \mathbb{R}^d$. The kriging model requires a set of unknown parameters to be estimated: θ, μ and σ^2. The maximum likelihood (ML) method is well adapted to estimate them.

Let the kernel function be defined by $k(\mathbf{x}, \mathbf{x}') = \sigma^2 r(\mathbf{x}, \mathbf{x}') = \sigma^2 r_{\mathbf{x}\mathbf{x}'}$, $\forall\ \mathbf{x}, \mathbf{x}' \in \mathbb{R}^d$, where $r_{\mathbf{x}\mathbf{x}'}$ is the correlation function between $\mathbf{x}$ and $\mathbf{x}'$ and the vector $\mathbf{r}_{\mathbf{x}\mathbf{X}} = [r_{\mathbf{x}\mathbf{x}^{(1)}}, \ldots, r_{\mathbf{x}\mathbf{x}^{(n)}}]^t$. In this work, we use the following Gaussian exponential kernel:

$$k(\mathbf{x}, \mathbf{x}') = \sigma^2 \prod_{i=1}^{d} \exp\left(-\theta_i \left(\mathbf{x}_i - \mathbf{x}'_i\right)^2\right) \ \forall \theta_i \in \mathbb{R}^+. \tag{2}$$

By initially assuming that θ-parameters are known, μ and σ^2 are given by a simple analytical expression

$$\hat{\mu} = \left(\mathbf{1}^t \mathbf{R}^{-1} \mathbf{1}\right)^{-1} \mathbf{1}^t \mathbf{R}^{-1} \mathbf{y}, \tag{3}$$

where $\mathbf{1}$ denotes an n-vector of ones, $\mathbf{R} = [\mathbf{r}_{\mathbf{x}^{(1)}\mathbf{X}}, \ldots, \mathbf{r}_{\mathbf{x}^{(n)}\mathbf{X}}]$ is the correlation matrix and

$$\hat{\sigma}^2 = \frac{1}{n} \left(\mathbf{y} - \mathbf{1}\hat{\mu}\right)^t \mathbf{R}^{-1} \left(\mathbf{y} - \mathbf{1}\hat{\mu}\right). \tag{4}$$

More details on the derivation of the equations (3) and (4) can be found in 25,26.

However, the d-vector of parameters θ remains to be estimated by maximizing the log-likelihood. Unfortunately, there is no analytical solution for estimating the vector θ like for the parameters μ and σ^2, an optimizer is thus necessary for this step. In fact, this step is the longest and the most fragile during the kriging building process since the dimension of the design space and the sample size are important. Moreover, high dimensionality leads to a long computation time and a very difficult parameter estimation process, harming the global quality of the kriging model. Indeed, the likelihood function is often multimodal and then it is inefficient to use a local optimization algorithm. A simple alternative consists of assuming that the objective function is the isotropic ($\theta_1 = \ldots = \theta_d$) and hence, change the optimization to one-dimensional. This significantly reduces predictive performances if isotropic assumption is not respected. In section C, we explain how to deal with this problem.

Finally, the best linear unbiased predictor for $y(\mathbf{x})$, given the observations $\mathbf{y}$, is

$$\hat{y}(\mathbf{x}) = \hat{\mu} + \mathbf{r}_{\mathbf{x}\mathbf{X}}^t \mathbf{R}^{-1} \left(\mathbf{y} - \mathbf{1}\hat{\mu}\right). \tag{5}$$

Moreover, the kriging model provides an estimate of the variance of the prediction, which is given by

$$s^2(\mathbf{x}) = \hat{\sigma}^2 \left(1 - \mathbf{r}_{\mathbf{x}\mathbf{X}}^t \mathbf{R}^{-1} \mathbf{r}_{\mathbf{x}\mathbf{X}}\right). \tag{6}$$

These two key equations can now be expressed in terms of the mean and variance and will be useful in the following to predict at any point the approximated value and its associated estimation error.

C. KPLS models for kriging in high dimension

As previously explained, the estimation of kriging parameters can be time consuming, especially for high dimensional problems. A recently developed method that accelerates the computational process while maintaining sufficient accuracy, consists of using the Partial Least Squares (PLS) method during the θ-estimation process. The resulting method is called KPLS (Kriging with Partial Least Squares).

The PLS method is a well-known tool for high dimensional problems that consists of maximizing the variance between input and output variables by the design variables in a smaller subspace, formed by the so-called latent variables. In the other hand, the parameters θ from the kriging model represent the range in any direction of the space. Assuming, for instance, that certain values are less significant, the corresponding θ_i will have a very small value relative to the others. In this way, the PLS method gives information on any variable contribution to the outputs and the idea developed in 10 consists of using this information to add weights on parameters θ. Latent variables computed through PLS method transcribe influence of initial variable to output ones and trim the dimension to h ($h << d$), the number of latent variables with h up to maximum 4. Hence, the spatial correlation $k(\mathbf{x}, \mathbf{x}')$ is replaced by a correlation function (Kernel) within which variables have a reduce dimension (h dimensions in total). Computational time is thus reduced and correlation matrix properties (symmetry and positivity) remain. More details of this method are given in 10.

D. KPLS-K models

In section C, we have seen that the KPLS method is able to rapidly build a kriging model for high-dimensional problems by reducing the number of θ-parameters. However, it seems in 10 that the KPLS model has some difficulties for approximating highly multimodal problems. In order to tackle this issue, a new approach is developed in 12, the so-called KPLS-K. It consists of adding a supplementary step during the construction of the KPLS model. In fact, this step is performed right after the estimation of the parameters of the KPLS model. However, an hypothesis is necessary for building the KPLS-K model which consists of assuming that kernels used are of exponential type (all Gaussian as Eq.(2)). This hypothesis allows us to pass from the KPLS model to the conventional kriging model. Through this transition, the dimensionality of the likelihood function changes from h dimensions to d dimensions, with $h << d$. Then, we make a local optimization of the kriging likelihood function (equivalent to the KPLS model) defined on the whole space by considering the solution θ provided by the KPLS model as a starting point. Thus, this approach searches a solution of the MLE (Maximum Likelihood Estimation) problem on a bigger search space and aims to improve the solution found by KPLS model. In fact, this method could be viewed as a new approach to estimate kriging parameters for high-dimensional problems through the KPLS method. The steps of such construction are given in the figure 1.

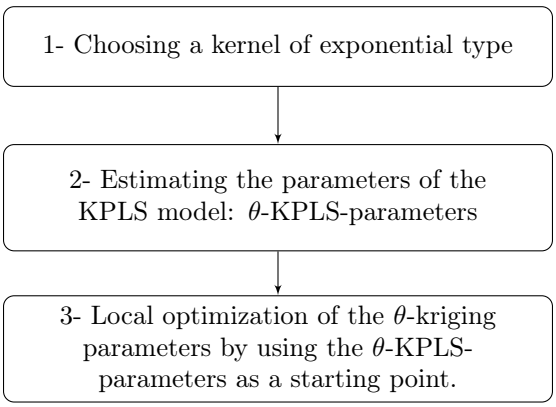


Figure 1. 3 main steps for constructing KPLS-K model.

E. Efficient Global Optimization algorithm

The Efficient Global Optimization (EGO) method,[3] is a bound constrained optimization method based on sequential enrichment of a surrogate model which approximates an expensive black box function to be

optimized. The kriging surrogate models are often used by this algorithm since they provide interesting statistical information. Indeed, they can predict the output function and also can provide the uncertainty of the model for any samples in the design space. Thus, it is possible to know where the output function is not well approximated by the model and we sometimes add samples in these areas. The uncertainty of the model on the training points is zero and the farther a point is from the training samples, the higher the uncertainty will be. These characteristics of the kriging model are shown in figure 2.

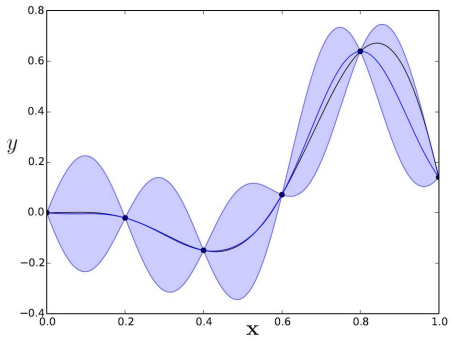


Figure 2. The kriging uncertainty at 99% of a black box function - The black line is the black box function, the blue line is the kriging function, the black dots are the training samples and the blue space is the uncertainty of the model.

Global optimization algorithms are said to balance exploration and exploitation. As mentioned above, EGO algorithm uses both the mean and the uncertainty provided by the kriging model. Indeed, the enrichment process typically balances exploitation and exploration behaviors:

- Exploitation is the search in regions that are close to good solutions, it is performed when we add a point in an area close to the current best solution $\mathbf{x}_{\min}$ that corresponds to

$$y_{\min} = \min(y^{(1)}, \ldots, y^{(n)}).$$

  By the way, we try to improve the accuracy of the model in the area containing $\mathbf{x}_{\min}$.

- Exploration is the search in regions that are sparsely sampled, it is performed when we add a point where the uncertainty is high. By the way, we try improving the accuracy of the model in the areas far from sampling points.

The most popular infill sampling criterion used by EGO algorithm is the expected improvement (EI) to be maximized (please see 3,5 for more details of such criterion). The EI function is given by

$$\mathrm{EI}(\mathbf{x}) = \begin{cases} (y_{\min} - \hat{y}(\mathbf{x}))\,\Phi\left(\frac{y_{min}-\hat{y}(\mathbf{x})}{s(\mathbf{x})}\right) + s(\mathbf{x})\phi\left(\frac{y_{min}-\hat{y}(\mathbf{x})}{s(\mathbf{x})}\right), & if\, s > 0 \\ 0, & if\, s = 0 \end{cases} \tag{7}$$

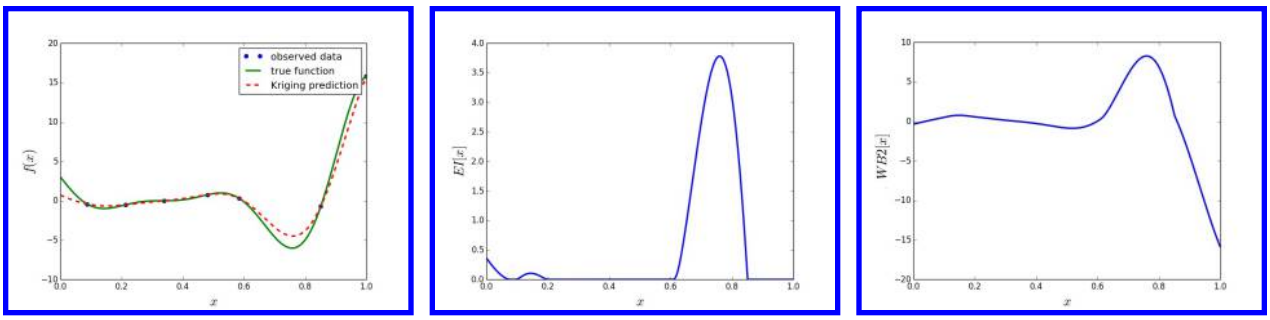
where $\phi(.)$ is the probability density function and $\Phi(.)$ is the cumulative distribution function of the standard normal distribution, $\hat{y}$ is the function to minimize in this case for instance a disciplinary output approximated by a kriging model and s^2 is its associated variance (respectively given by Eq (5) and Eq (6)). The expression of EI is a balance between seeking promising areas of design space and the uncertainty in the model. The term relative to $\Phi(.)$ is large when $\hat{y}$ is small with respect to $y_{\min}$ and then promotes exploitation. The term relative to $\phi(.)$ is large when $s(x)$ is large and then promotes exploration.

Figure 3 shows a 1-dimension example of how the EI criterion works. The maximum of EI function locates the next sampling point, see figure 3-b.

In this study, the WB2 criterion, to be maximized and given by equation (8), is chosen.

$$\mathrm{WB2}(\mathbf{x}) = -\hat{y}(\mathbf{x}) + \mathrm{EI}(\mathbf{x}) \tag{8}$$

As for the EI criterion, the maximum of WB2 function locates the next sampling point, see figure 3-c. In fact, M. Sasena[5] advised to use the WB2 criterion after comparing several infill sampling criteria with many



(a) 1-D function. (b) EI criterion. (c) WB2 criterion.

Figure 3. The EGO enrichment for a black box function (1D function $f(x) = (6x-2)^2\sin(12x-4),\ x \in [0,1]$) with the EI and WB2 criteria. (a) The green line is the true function, the red line the predicted kriging model interpolating the observed data (7 blue points). (b) The EI function when only 7 points have been sampled. (c) The WB2 function when only 7 points have been sampled seems to be smoother.

test examples. Moreover, WB2 function is more local and smoother than the EI criterion. The former characteristic is very interesting for high dimensional problems since the uncertainty will be very high on these cases. Thus, WB2 function is a good criterion in order to avoid too much enrichment in areas where the uncertainty is very high. This criterion is easier to optimize than the EI one thanks to its smoother properties as illustrated on figure 3-c. In addition, it helps the algorithm to rapidly converge to a solution, even if it is a local solution, since the number of sampling points is generally limited.

F. SuperEGO algorithm

In section E, we have seen the EGO method, which is adapted to bound constrained optimization problems. In order to take constraints into account, the SuperEGO (denoted by SEGO in the following) approach was developed by M. Sasena.[5] Indeed, the aircraft design optimization problems are often constrained as explained previously. Thus, SEGO maximizes the WB2 criterion subject to the constraints thanks to an optimizer which takes them into account. The optimizers (COBYLA, SLSQP, NSGA2) will be used to maximize the WB2 criterion (8). The optimization problem becomes

$$\begin{cases} \max_x \mathrm{WB2}(\mathbf{x}) \\ s.t. \\ \hat{c}_1(\mathbf{x}) \leq 0 \\ \vdots \\ \hat{c}_m(\mathbf{x}) \leq 0 \end{cases} \tag{9}$$

where $\hat{c}_i$ is the prediction of the constraint c_i, m is the number of constraints and $\mathbf{x} \in \mathbb{R}^d$. The optimizer could be selected from the libraries SciPy,[27] pyOPt[28] or pyOptSparse.[29] The most commonly used are:

COBYLA for Constrained Optimization BY Linear Approximation is an implementation of Powell's non-linear derivative–free constrained optimization that uses a linear approximation approach. More details are given in 13.

SLSQP for Sequential Least Squares Programming optimizer is a sequential least squares programming algorithm which uses the Han–Powell quasi–Newton method. This algorithm[30] based on gradient method uses the Jacobian calculation of the objective and the constraints function.

NSGA2 for Non Sorting Genetic Algorithm II is a non-dominating sorting genetic algorithm that solves non-convex and non-smooth single and multiobjective optimization problems. The algorithm attempts to perform global optimization, while enforcing constraints using a tournament selection-based strategy.[31]

Finally, the summary of SEGO algorithm steps is given in figure 4.

In order to handle high-dimensional optimization problems with a large number of constraints, some proposed improvements of the SEGO algorithm are described in the following. The main objective is still to minimize the number of calls to expensive black box.

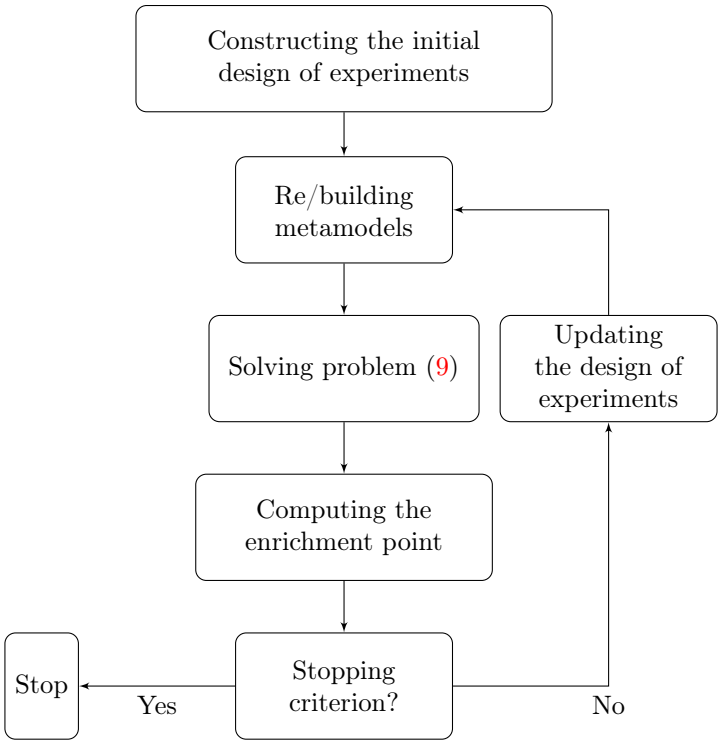


Figure 4. The summary of SEGO algorithm steps.

IV. Improvements of efficient global optimization

In order to minimize the number of calls to the expensive black box functions relative to the objective function and the constraint functions, we aim at approximating with sufficient accuracy the behaviour of these functions from few sampling points. Some improvements proposed concern thus the use of more complex surrogate models able to give a accurate prediction with its associated variance. This new kind of surrogate models will be combined in the SEGO algorithm and the resulting algorithm will be integrated in a framework dedicated to a multi-disciplinary optimization.

A. Overview of mixture of experts

As already mentioned previously, actual strategy to reduce significantly computational time consists in building response surfaces, called meta-model. This response surface needs to mimic accurately the objective function for minimal simulator calls. By optimizing the meta-model instead of the objective function, it is possible to save computational time. This remark is also valid for the constraints which can be expensive computer codes (buckling factor for structure, drag for aerodynamic, ...). There exist a lot of different ways to build a meta-model: in the case of simple dependencies, linear or polynomial models can be used but when interactions become more complex, elaborated model as neural network, radial basis function, or Gaussian process regression as kriging are required. Another idea consists in using multiple surrogate models as local experts to approximate different parts of the input space and to combine them in an automatic way in order to possibly decrease the errors. 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, in order to have several surrogate models to be responsible for different parts of the input space, to enable modeling the heterogeneous complexity in the function profile ("Divide and Conquer" principle). A general introduction about the mixture of experts can be found in 32 and a first application with generalized linear models in 33. More generally, mixture of local experts can be build in order to increase the accuracy of the global model by combining automatically some of the local experts listed previously. Based on this idea and a previous paper from the present authors,[6] a python toolbox called MOE (Mixture Of Experts) has been developed and will be explained in the following.

This method is intended to improve the accuracy of the approximation for functions with some of the following characteristics: heterogeneous behaviour depending on the region of the input space, flat and steep regions, first and zero order discontinuities (as for instance buckling in structural mechanics[34]). It strongly relies on the Expectation-Maximization (EM) algorithm for Gaussian mixture models (GMM). With an aim 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, cubic, radial basis functions, or different forms of kriging) on each cluster and all the local experts are finally combined using the Gaussian mixture model parameters found by the EM algorithm to get a global model. In this approach, the Gaussian mixture model is used with conjoint data both to partition the input space and to derive the mixing proportion. The approach developed in 35 is based on two learning algorithms (an unsupervised algorithm and a supervised one): the first one to find the clustering and the second one to compute the corresponding cluster posterior probability, which will be used to evaluate the mixing proportions. As explained in 6, we must predict outputs from inputs to create a model. In our case, outputs are scalars $y_i \in \mathbb{R}$ and inputs are vectors $\mathbf{x}^{(i)} \in \mathbb{R}^d$. To perform the clustering, we need n inputs $\mathbf{X} = (\mathbf{x}^{(i)})_{1\ldots n}$ and n outputs $\mathbf{y} = (y^{(i)})_{1\ldots n}$. So we can only know the cluster posterior probabilities of vectors like $(\mathbf{x}^{(i)}, y^{(i)}) \in \mathbb{R}^{d+1}$. To predict the cluster posterior probabilities of a sample knowing only its inputs, we must project each multivariate the Gaussian functions k of Gaussian mixture model, trained in dimension $d+1$, on the inputs hyperplane of dimension d. Thus for each cluster k, we create a multivariate Gaussian function in dimension d with the covariance matrix $\mathbf{\Gamma}_k^X$ and the mean vector $\boldsymbol{\mu}_k^X$.

$$\mathbf{\Gamma}_k = \begin{pmatrix} \mathbf{\Gamma}_k^X & \nu_k \\ \nu_k^T & \xi_k \end{pmatrix}, \tag{10}$$

where $\mathbf{\Gamma}_k$ is the covariance matrix of $(\mathbf{X}, \mathbf{y})$, $\mathbf{\Gamma}_k^X \in \mathbb{R}^d$ is the covariance matrix of $\mathbf{X}$, $\nu_k \in \mathbb{R}$ is $\mathrm{Cov}(\mathbf{X}, \mathbf{y})$ and $\xi_k \in \mathbb{R}$ is $\mathrm{Var}(\mathbf{y}, \mathbf{y})$

$$\boldsymbol{\mu}_k = \begin{pmatrix} \boldsymbol{\mu}_k^X \\ \mu_k^y \end{pmatrix}, \tag{11}$$

where $\boldsymbol{\mu}_k$ is the mean vector, $\boldsymbol{\mu}_k^X$ is the X-coordinates of the mean $\boldsymbol{\mu}_k$ and μ_k^y is the y-coordinates of the mean $\boldsymbol{\mu}_k$. Thanks to hyperplane projection and linear recombination, inputs cluster posterior probabilities of each cluster can be predicted and local models can be performed. When local models f_i are known, the global model would be

$$\hat{f}(\mathbf{x}) = \sum_{i=1}^{K} \mathbb{P}(\kappa = i | X = \mathbf{x}) \hat{f}_i(\mathbf{x}) \tag{12}$$

which is the classical probability expression of mixture of experts. In this equation (12), K is the number of Gaussian components, $\mathbb{P}(\kappa = i | X = \mathbf{x})$, denoted by gating network, is the probability to lie in cluster i knowing that $X = \mathbf{x}$ and $\hat{f}_i$ is the local expert built on cluster i.

Eq. (12) leads to two different approximation models depending on the computation of $\mathbb{P}(\kappa = i | X = \mathbf{x})$. When choosing the Gaussian laws to compute this quantity, Eq.(12) leads to a **smooth model** that smoothly recombine different local experts. If $\mathbb{P}(\kappa = i | X = x)$ is computed as characteristic functions of clusters (being equal to 0 or 1) this leads to a **discontinuous approximation model**.

B. Proposed approach for mixture of experts

We improve mixture of experts in order to perform the best surrogate models. Different approaches have been made concerning the choice of the clustering criterion, the choice of the number of clusters, the choice of local models, the choice of the scale factor, the choice for high dimensional problems, the computation of the Jacobian and the estimation of the error. These different points are explained in the following. In order to assess the proposed improvements, we compare results on 1-Norm function in d dimension, which is derivative-discontinuous function:

$$\text{1-Norm} = \|\mathbf{x}\|_1 = \sum_{i=1}^{d} |x_i| \text{ where } d \in \mathbb{R}^*, \quad \mathbf{x} \in \mathbb{R}^d \tag{13}$$

with the following error criteria: Mean Square Error (MSE) and Lack of Fit (LOF)

$$\text{MSE} = \frac{1}{n_v}\sum_{i=1}^{n_v}(y_i - \hat{y}_i)^2; \quad \text{LOF} = \frac{100\frac{1}{n_v}\sum_{i=1}^{n_v}(y_i - \hat{y}_i)^2}{\text{Var}(\mathbf{y})} \tag{14}$$

where n_v denotes the number of validation points, $\mathbf{y} \in \mathbb{R}^{n_v}$ is the output vector of components y_i and $\hat{\mathbf{y}}$ the predicted ouput vector of components $\hat{y}_i$. The main interest of this function is that it features several and clearly distinct linear parts, being the intersection of 2^d hyperplanes. In the following tests, the validation dataset is given by 10% of the database to compute the error criteria. In the following sections, we propose some experiments firstly with $d = 2$ for illustrative purpose. Then we extend to high-dimensional problem with $d = 80$. Figure 5 illustrates the 1-Norm function in 2D and its associated clusters.

(a) Experimental samples in 2D.

(b) Gating network for 1-Norm $d = 2$ with 5 clusters.

Figure 5. 2D illustrative example for 1-Norm - samples and clustering.

1. *Clustering criteria*

In the initial version,[6] the clustering was based on spatial location and function values (supervised approach between the inputs $\mathbf{X}$ and output $\mathbf{y}$). If available, some other clustering criteria could be used such as derivative value (supervised approach between the inputs $\mathbf{X}$ and outputs $\partial\mathbf{y}/\partial\mathbf{x}$) in order to get a better indicator of the heterogeneity in the function profile as mentioned in 35. In the developed toolbox, the user can provide the Jacobian or some relevant components of the Jacobian as a criterion for the clustering. Table 1 gives the different criteria available to perform the clustering and the associated best number of clusters which has been found following the strategy described in the next section 2.

Function	Clustering Criteria	Number of Clusters
1-Norm $d = 2$	y	5
	Jacobian $(\partial y/\partial x_1, \partial y/\partial x_2)$	4
	First Jacobian Component $(\partial y/\partial x_1)$	4
	Second Jacobian Component $(\partial y/\partial x_2)$	4

Table 1. 1-Norm ($d = 2$) - Variability of number of clusters with clustering criteria.

2. *Choice of the number of clusters*

One of the main challenges in MOE modeling is the automatic determination of the number of experts a priori, which has been identified as a difficult problem in data clustering in general. Here, the number of clusters is chosen through a range of potential clusters by minimizing the generalization error for the mixture of linear (quadratic or kriging) experts. We can find the best number of clusters to improve mixture of experts. For best results, some rules of thumb have been carried out.

- The number of clusters cannot be higher than 10% of the number of training samples.

- For each clustering, we estimate the errors through a well-known cross-validation procedure with the classical partition $(4/5, 1/5)$ for learning basis and test basis. The two errors presented here are the mean square error and the median of mean square errors of cross-validation.

Figure 6 illustrates the evolution of these both error criteria in order to determine automatically the optimal number of clusters minimizing the errors. In this case, some quadratic models have been imposed as local experts to compute the errors quickly, 5 clusters seems to be the optimal value here. Moreover, for each number of clusters, errors are computed with a smooth mixture of models or with a discontinuous global model for comparison. The algorithm chooses automatically the number of clusters that minimizes the errors.

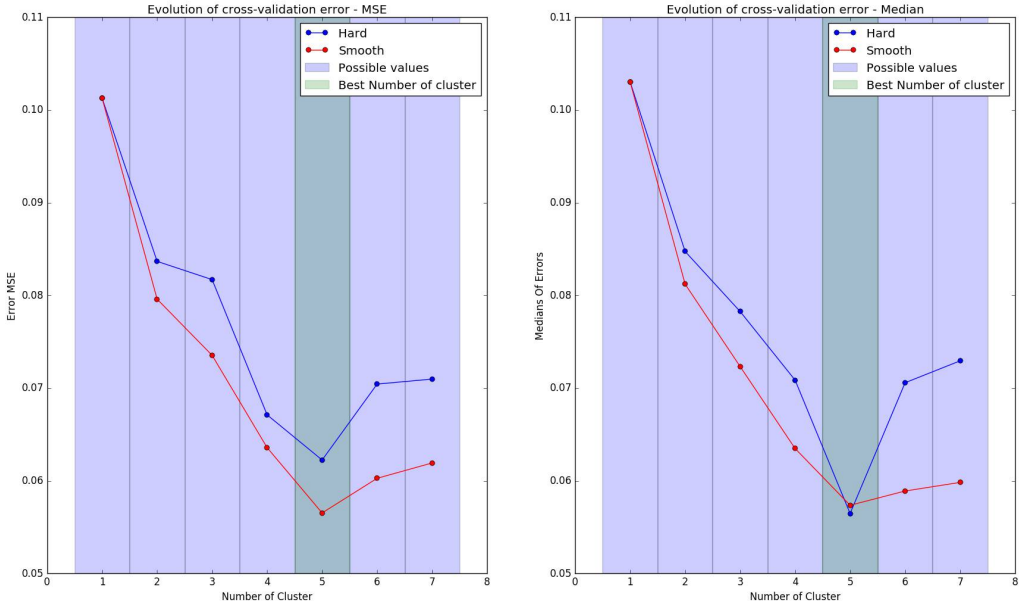


Figure 6. 1-Norm ($d = 2$) - Evolution of cross validation error by models: the blue columns show that the number of clusters is feasible, the white column shows that the number of clusters is not feasible (there are one or several clusters which are empty) and the green column shows the best number of clusters. Some quadratic models have been used here as local experts because they are pretty cheap in comparison for instance with kriging.

In case there are several number of clusters that could do it, we would better choose the lowest to make the clusters as big as possible to build an accurate local expert.

3. Choice of local models

Different local experts have been considered:

some surrogates from the Scikit-learn toolbox[36] : Least Square Model (LS), Square Polynomial Regression Model (PA2), Cubic Polynomial Regression Model (PA3), Ordinary Kriging (OK) model with square exponential correlation, Radial Basis Function Model (RBF). More details on these models can be found in 25.

some "in house" surrogate models : Multi-Fidelity Kriging (MFK),[37,38] Kriging Partial Least Square (KPLS),[10] Improved KPLS models (KPLS-K).[39] Concerning the MFK model, when only one DOE is given as inputs (one fidelity is assumed in this case), the model corresponds to an ordinary kriging model where the kernel can be chosen. The algorithm considers different correlation kernels and chooses the one where the error (MSE) is minimized (squared exponential kernel, absolute exponential kernel, Matérn $\frac{3}{2}$, Matérn $\frac{5}{2}$, more details on these correlation functions can be found in 21).

From these different local experts, a smooth recombination or a discontinuous one can be performed. In general the two computations are done and we keep the one with lowest error. In table 2, we report error criteria for different number of clusters associated with different type of surrogate models.

Function	Number of Clusters	Models	MSE	LOF
1-Norm $d = 2$	1 imposed	LS imposed	1.77E-01	1.00E+02
	1 imposed	PA2 imposed	1.15E-02	6.51E+00
	5 imposed	PA2 imposed	3.76E-03	1.73E+00
	5 imposed	PA3 imposed	5.08E-03	2.33E+00
	6 imposed	LS imposed	7.60E-04	3.49E-01
	6 imposed	PA2 imposed	3.51E-03	1.61E+00
	6 imposed	PA3 imposed	4.96E-03	2.28E+00
	6 imposed	3LS/MFK/KPLS-K/OK	3.50E-04	1.61E-01
	5	3LS/MFK/OK	1.63E-04	7.47E-02

Table 2. 1-Norm ($d = 2$) - Mean Square Errors and Lack of Fits with different number of clusters (imposed number or best number) and different models (imposed models or best local models).

Table 2 shows that the best mixture is obtained by combining the best number of clusters (5 selected clusters) and the best local models (5 selected experts: 3 LS models, 1 MFK model and 1 OK model). On this example, the MSE and LOF criteria are both minimized with the automatic choice given by the MOE process. As expected with the 1-Norm function, the mixture of 6 linear experts (LS) gives a relatively small error (7.60E-04) compared to mixture with polynomial experts of higher degree (PA2 or PA3). Moreover, even if the local expert is very simple as LS or PA2 for instance, it could be more interesting to consider several clusters rather a unique one as seen in table 2 when PA2 is imposed to 1, 5 or 6 clusters. These results validate the strategy developed to choose automatically the number of clusters and the local experts.

4. Choice of the slope factor

In order to reduce the error of the smooth recombination, the gradient of the multivariate Gaussian functions representing the clustering can be modified by a weighting factor: the covariance matrix of multivariate Gaussian Functions is multiplying by this factor (denoted by the slope factor or the Heaviside factor) and the slope is changing around the cluster boundary. A search of the best factor is done in the interval $[0.1, 2.1]$. Results show that a well-chosen slope factor can improve the mixture of experts as seen in figure 7 where the optimal value is given by a factor equals to 0.4. On the contrary, a bad Heaviside factor increases the mean square error (factor equals to 2 for instance on figure 7).

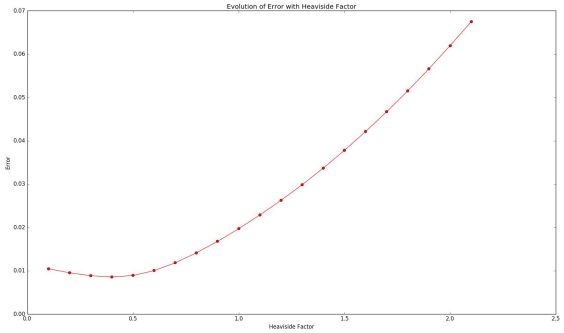


Figure 7. Evolution of the error as the Heaviside factor values.

5. *Choice for High Dimensional problems*

In order to test our MOE in high dimension, the parameter d relative to the dimension of 1-Norm function (13) is increased: 20, 40 and 80 dimensions are considered. To reduce the computation time corresponding to the choice of the optimal number of clusters, a RBF surrogate model is used to test the different numbers of clusters, this surrogate is cheaper than quadratic regression at high dimension.

Different combinations have been done to reduce the computational cost of each MOE building step. Some examples of MSE errors are reported in table 3. These results prove that errors are to the minimum reduced when the stage for the search of the best local models is carried out.

Dimension	Number of samples	Clustering Model	Local models	MSE
20	400	RBF	Best	2.19E-02
40	800	RBF	Best	4.20E-02
80	800	RBF	Best	2.34E-02
20	400	RBF	KPLS3	1.68E-01
40	800	RBF	KPLS3	2.92E-01
80	800	RBF	KPLS3	5.85E-01

Table 3. MSE for mixture of experts with RBF clustering model and different local experts (Best choice or imposed choice with KPLS with $h = 3$ latent components) for 1-Norm function in dimension 20, 40 or 80.

The automatic strategy is still under development but theoretically well adapted to discontinuous functions (C^0 or C^1).[34]

6. *Saving some computational time*

If some *a priori* information is known about the function to approximate in terms of complexity, regularity,... it is also possible to impose the number of clusters and/or the type of local experts. Indeed the previous steps described to choose the optimal parameters (number of clusters, best local experts, slope factor for the smooth recombination, ...) could be time consuming and the user can impose some choices to save some computational time. For example, the option to consider a single cluster with a given expert is the obvious one.

7. *Analytical Jacobian of the Mixture of experts*

Moreover, the Jacobian of the global model can be predicted if the Jacobians of the local models are known. Thus if a smooth recombination is chosen, we can perform a gradient optimization thanks to the smoothness of the surrogate model. Jacobians of some local experts (linear, quadratic, cubic, and the different forms of kriging) have been computed analytically in order to have the Jacobian of the mixture of experts for gradient based optimizers as the one used in the following section (SLSQP optimizer for instance).

8. *Error estimation for the Mixture of experts*

For some models like the kriging-based models, we are able to predict the uncertainty of the predicted values. When a smooth recombination is considered, we suppose that the uncertainties of these models follow Gaussian functions. Thus, if all of the local models of the mixture can predict their uncertainty, the mixture of experts can also predict its uncertainty if we assume that they are independent random variables that are normally distributed (and therefore also jointly so). Then their sum is also normally distributed:

$$\sum_k \alpha_k \mathcal{N}(\hat{y}_k, s_k^2) = \mathcal{N}\left(\sum_k \alpha_k \hat{y}_k, \sum_k \alpha_k^2 s_k^2\right). \tag{15}$$

For a hard recombination, we do not have to suppose that uncertainties of models follow Gaussian functions. Indeed, for one sample, we choose the uncertainty corresponding of its cluster. Thus, the mixture of experts can also predict the uncertainty. This information is all the more interesting as it can be used in the surrogate-based optimizers like EGO or SEGO.

C. SEGOMOE

As described in section F and on figure 4, the SuperEGO problem finds the optimum of the objective function by enriching the design of experiments. It takes into account the estimation itself and the estimation of the variance made by the surrogate model. As the mixture of experts can provide this information when local experts issued from kriging (OK, MFK, KPLS, KPLS-K) are used, the SEGOMOE algorithm has been developed to combine, on the one hand, the accuracy given by the MOE and, on the other hand, the compromise between exploration and exploitation with the SuperEGO algorithm.

Figure 8 illustrates the SEGOMOE optimization process using an XDSM (eXtended Design Structure Matrix) diagram.[40] The thin black lines and thick gray lines in the XDSM diagram represent the process flow and the data dependencies, respectively. For more details on the XDSM diagram conventions, see Lambe and Martins.[40]

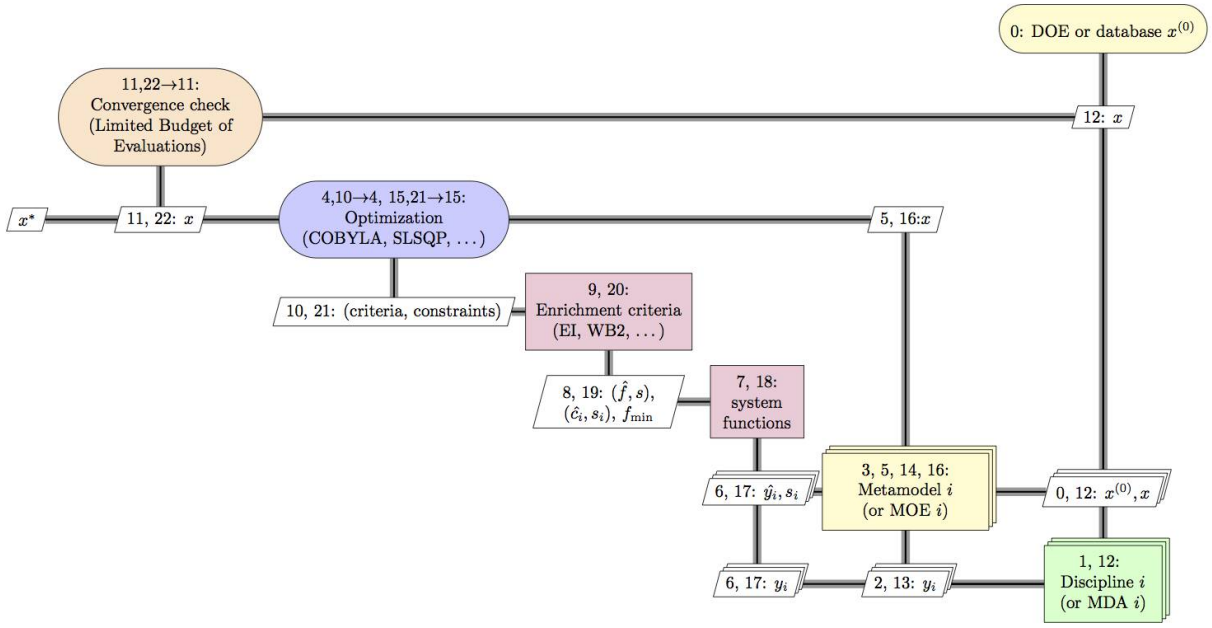


Figure 8. SEGOMOE algorithm

The Algorithm in appendix A on page 25 explains in details the process of this combined optimization method. As SEGO, the main steps are the following:

- Constructing the initial database $(\mathbf{X}, \mathbf{y})$ and build the associated MOE models relative to the objective function and the constraint functions.
- Solve the optimization problem with the WB2 (or EI) criterion (8) to maximize under constraints and propose the new enrichment point. The both information $(\hat{y}(\mathbf{x}), s(\mathbf{x}))$ required to compute these criteria, are given by a mixture of experts from Eq. (15).
- Compute the values of the objective function and the constraint functions at the new enrichment point and update the associated MOE models.
- If the stopping criterion is not reached, go back to the optimization problem to propose a new enrichment point.

SEGOMOE iterates until the stopping criterion is met. Due to the high computational cost of actual simulations, it is common to use the maximum number of function evaluations as the stopping criterion. Another alternative is to set a target value for the expected improvement (or WB2 criterion). The proposed method SEGOMOE could be compared to the method proposed in 41 based on the simultaneous use of multiple surrogates in order to add several points per optimization cycle.

Concerning the Optimization process, more details can be added about some of these steps (the numbering is referring to appendix A on page 25).

- At step 0, if no point $\mathbf{x}^{(0)}$ is given by the user to initialize the optimization algorithm, we create a DOE of the number of chosen samples. In general, if dimension d is considered, an initial DOE with $d+1$ sampling points is created.

- For the training of the MOE, we impose kriging (OK, MFK), KPLS, or KPLS-K as local experts for the objective function in order to get the associated error estimation. Indeed, the SEGOMOE algorithm requires to have both the mean and the uncertainty to compute the WB2 (or EI) criterion relative to the objective function. The key point here is the use of Eq. (15) to get this both information from the mixture of experts. As the SEGOMOE algorithm uses only the approximated value of constraint functions (denoted by $\hat{c}_i(\mathbf{x})$), no restriction is done for the constraints and all of the available surrogates could be used (PA2, PA3, ...).

- During the enrichment process, the optimal number of clusters and the best model for each cluster are updated every κ iterations, where κ is a fixed number.

- In order to save computational time, the user can impose some choices as mentioned in 6. In particular, when the number of outputs (objective function and constraints) is too large, the time to build all the surrogate models (the best number of clusters, the best local experts, ...) is very costly and the update every κ iterations can be skipped or κ has to be well chosen.

- The initialization (steps 4 and 15) consists of finding the best starting point in the training samples violating the fewer constraints and minimizing the objective function f of problem (1). For this purpose, the sum of violated constraints is computed on each sample and the one minimizing it is selected.

- The best enrichment sample is found thanks to the COBYLA, SLSQP or NSGA2 optimizer. Note that SLSQP and COBYLA use a multistart optimization (10 different starting points). If available, the algorithm SLSQP will use the analytical Jacobian of the mixture of experts (as described in section 7) to compute the gradients of both the objective function (EI or WB2) and the constraints of the global optimization problem (9).

- The main optimization loop stops when the maximum number of evaluations is reached. The best point $\mathbf{x}^\star$ is selected among the database as the sampling point violating less constraints (by comparing the sum of violated constraints) and minimizing the objective function f of problem (1).

D. Integration

In order to use MOE surrogate model and SEGO optimizer conveniently within a multi-disciplinary optimization process, they were integrated within the OpenMDAO framework developed by the NASA Glenn Research Center.[42,43] That open-source framework written in Python aims at enabling the user to assemble models and solve coupled derivatives efficiently[44] in order to carry out multi-disciplinary optimization. A key property of that framework is that it can be extended to integrate new numerical methods.

The integration of MOE as a surrogate model is rather straightforward in OpenMDAO as the framework owns a dedicated `SurrogateModel` interface. That interface exposes typical `train()` and `predict()` methods that map almost immediately to the native interface of the MOE module.

The SEGOMOE optimizer integration required more work but was facilitated by the usage of the common interface introduced by the pyOpt[28] framework. Indeed, pyOpt was created to standardize the programmatic access to various optimizers in order to be able to switch them with less client code change as possible. While pyOpt is still usable, the project web site does not seem to be active anymore (the last 1.2 version was released mid-2014). Actually, a spin-off project, pyOptSparse[29] appears to be the replacement project. It is a fairly extensive (not backward compatible) modification of the original project pyOpt while keeping the basic idea of a unique way of accessing multiple optimizers. OpenMDAO uses pyOptSparse now and makes the optimizers available through a dedicated driver object implementing the framework `Driver` interface.

To facilitate further usage even outside OpenMDAO, SEGOMOE interface was first adapted to match pyOptSparse conventions. Then, a dedicated driver was implemented like the pyOptSparse driver to handle the SEGO optimizer integration within OpenMDAO as illustrated on figure 9. Further integration would require to propose our SEGO adapter layer directly within pyOptSparse framework, allowing to remove the specific implementation of the driver interface, `SEGODriver`.

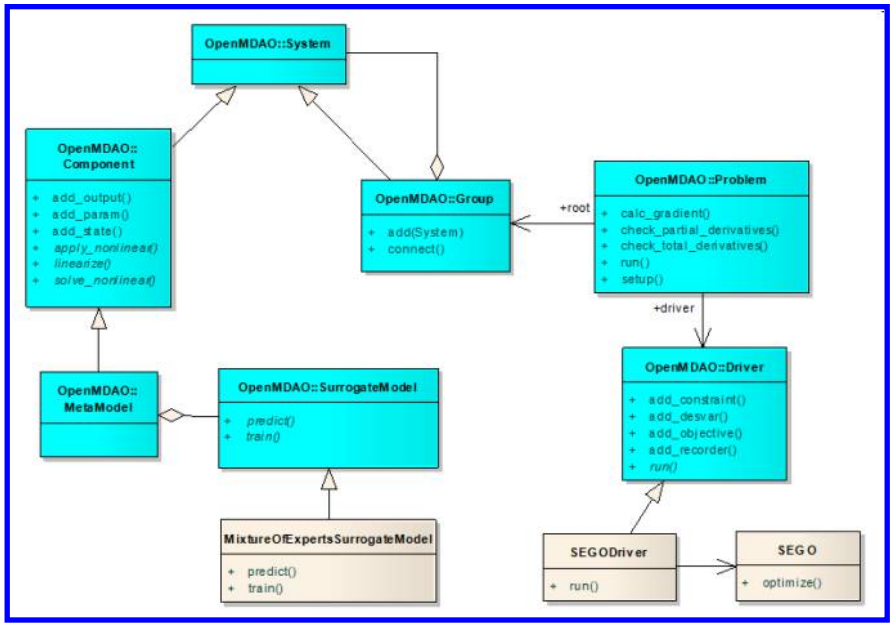


Figure 9. MOE surrogate model and SEGO optimizer integration in OpenMDAO.

V. Numerical results

This section is dedicated to validate SEGOMOE on several test cases (academic functions and MOPTA). We also want to highlight our very first results on wing design optimization.

A. Analytical results on academic functions

In order to validate the efficiency of this optimization method, it should be assessed on reference tests cases whose results are known. In that aim, five problems have been selected: four academical functions (described in section B on page 26) and an automotive industrial test case. The four test cases based on the academical functions, described by the table 4 followed the same experimental process as the one used in 39, 45.

Function	d	m	Domain definition	Optimum or best known value[39, 45]
PVD4	4	3	$[0.1]^2 \times [0, 50] \times [0, 240]$	5804.45
g_{02}	10	2	$[0, 10]^{10}$	-0.8
g_{07}	10	8	$[-10, 10]^{10}$	24.3062
g_{10}	8	6	$[10^2, 10^4] \times [10^3, 10^4]^2 \times [10, 10^3]^5$	7049.3307

Table 4. Academic functions tested. d is the dimension number and m is the number of constraints. Please see appendix B for the expressions of these functions.

In order to have information about the stability of SEGOMOE, the following choices are made:

- Each test is repeated 30 times.
- A design of experiments (LHS with ESE criterion) of $d + 1$ samples is made, so it is different for each of the 30 tests.
- Some mixtures of experts are built automatically for the objective function and the associated constraints in order to make the best choices (number of clusters, local expert, slope factor for the smooth recombination, ...).

- 100 SEGOMOE iterations are made and the best value found is saved and kept to be reported in the table.
- The COBYLA[27] optimizer is used for the enrichment step of the MOE.

For each of these functions, the best solution, the worst solution, the median, the mean and the standard deviation are given in table 5 and compared with the results obtained in 39,45. In fact, several optimization algorithms are used in 45 and the best results of each case test are retained. One should note that the SEGOKPLS algorithm, developed in 39, is a particular case of SEGOMOE: it consists in imposing a single cluster with a single expert given by a KPLS model for each surrogate which is to be built. Considering the

Function	PVD4	g_{02}
Best known value	5804	-0.80
Best	(**5804.61**, 5848.66, 5807.79)	(**-0.44**, -0.30, -0.33)
Worst	(6183.52,**6179.71**, 7669.36)	(-0.18,**-0.19**, -0.16)
Median	(**5824.27**, 5959.71,6303.98)	(**-0.26**, -0.23, -0.20)
Mean	(**5867.53**, 5960.54, 6492.45)	(**-0.26**, -0.23, -0.21)
Standard deviation	(90.75, **75.66**)	(0.06, **0.04**)
Function	g_{07}	g_{10}
Best known value	24.30	7049.33
Best	(**24.30**, **24.30**, 24.48)	(**7289**, 8210.08, 7818.53)
Worst	(24.95, **24.30**, 1514.98)	(17490, **11285**, 13965.60)
Median	(**24.30**, **24.30**, 25.28)	(12409, 9957.84, **9494.06**)
Mean	(24.33, **24.30**, 25.35)	(12372, **9702.63**, 10018.33)
Standard deviation	(0.12, $5.51e^{-6}$)	(2872, **1647**)

Table 5. Statistical results of the best points of the objectives functions tested with 30 repeats for 100 optimization iterations. Results in parenthesis represent: on the left, the results of SEGOMOE, on the middle, the results of SEGOKPLS,[39] on the right, the best results obtained by the algorithms tested by R. Regis 45. The best results are in bold and in underlined green if they correspond to the best known value. For standard deviation: on the left, the results of SEGOMOE, on the right: the results of SEGOKPLS.

table 5, the optimization with the SEGOMOE algorithm can reach a better solution than the SEGOKPLS optimizer. Furthermore, it must be noticed that the means of the results of the optimization are also better for the test case $g02$, $g07$ and PVD4. Indeed, the $g10$ test case shows a worst mean than the SEGOKPLS optimization process. The optimum is difficult to reach because the function is very sensitive near the optimum. So even with the action of the MOE surrogate model, the optimum region is difficult to model. Regarding the standard deviation of these several test cases, it appears that the SEGOMOE optimizer is less stable than SEGOKPLS. As explain earlier, the process build different DOEs at the beginning. Thus, the standard deviation results show that the SEGOMOE optimizer seems more dependant of the information which is available at the beginning of the optimization process. The different choices made to build automatically the mixture of experts would require to be more robust in order to minimize this deviation. These first results validate the SEGOMOE approach as a global optimizer handling constraints up to dimension 10.

B. Results on MOPTA test case

MOPTA test case which comes from automotive industry[46] and can be downloaded at http://www.miguelanjos.com/jones-benchmark.[14] This optimisation problem aims at reducing the weight with constraints of crash, vibration and mechanical. There are 124 inputs and 69 outputs which correspond to 1 objective function and 68 constraints. This is a kriging model based on an expensive simulation model that can take 1 to 3 days to evaluate. The objective here is to optimize this problem with the SEGOMOE algorithm, the known solution of this weight minimization being **222.23 kg**. Instead of optimize directly the complete problem with 124 inputs and 69 outputs, the dimension is reduced to 12. In fact, this reduction

can be done because we already know the best optimum found by R. Regis in[45] and some inputs can be fixed to the optimum. Furthermore, two different optimizers have been included in the enrichment process: COBYLA and SLSQP, both using a multistart optimization with 10 different starting points. As SLSQP is a gradient based optimizer, if available the analytical derivative of the MOE will be used.

The test follows the given process:

- Each test is repeated 5 times.
- Two stopping criteria are imposed. The first one is the maximum number of iterations that the algorithm SEGOMOE will be permitted to perform (200 iterations). As in this particular test case, the solution is known, the second criterion concerns a relative error with a threshold of 10^{-6} between the known solution and the best current value.
- A design of experiments of $d+1$ samples is made with d the dimension of the inputs. This DOE is an optimized LHS with the ESE criterion as described previously.
- Some mixture of experts are built automatically for each of the 69 outputs (1 objective function and 68 constraints) and updated at the end of each SEGOMOE iteration.
- The number of calls to the black box is reported and in the MOPTA case one call to the simulator yields all the values for $f(\mathbf{x}), c_1(\mathbf{x}), \ldots, c_m(\mathbf{x})$.

For each dimension tested for the MOPTA test case, only the number of calls of the black box function is taken into account corresponding to the number of points for the DOE sampling and the number of iterations to reach the convergence. Table 6 reports the maximum number of calls required to get the reference optimum with the 5 runs. The SEGOMOE algorithm will be compared to the SEGOKPLS and SEGOKPLS-K algorithm developed in 39. It is based on the EGO algorithm taking constraints into account. The kriging model (KPLS or KPLS-K) used is designed to deal with high dimension problem thanks to a partial least square approach and corresponds to the SEGOMOE approach restricted to a single cluster. Some comparisons are alse made with a direct approach based on COBYLA algorithm without any surrogate model. For this purpose, COBYLA through the OPTI toolbox[47] that runs in Matlab has been used. In this case, 3 runs over 5 have converged and the worst value in terms of number of simulation calls is equal to 500.

Function	SEGOMOE		SEGOKPLS	SEGOKPLS-K	COBYLA
	COBYLA	SLSQP	COBYLA	COBYLA	DIRECT
MOPTA 12D	39(13+26)	40(13+27)	62(13+49)	52(13+39)	> 500

Table 6. MOPTA 12D - Number of calls (worst value of the 5 runs) of the black box function to reach the optimum (DOE samples + number of iterations) with SEGOMOE, SEGOKPLS or SEGOKPLS-K approaches. Comparisons are made with COBYLA algorithm on a direct approach (optimization directly done on the MOPTA code).

The results of the table 6 show that the SEGOMOE needs less calls to the black box function to reach the optimum. So the convergence of the SEGOMOE is faster than the SEGOKPLS process. To see the impact of the SEGOMOE optimizer on higher dimensional problem, we consider the same test case with 30 design variables. Table 7 reports the maximum number of MOPTA evaluations (worst value of the 3 runs) to converge to the reference solution. As the dimension equals to 30, we impose in such case a mixture of local experts based only on KPLS or KPLS-K models in order to reduce the CPU time for the choice of the best local experts. The number of calls is still very few relative to the dimension considered.

Function	SEGOMOE (KPLS or KPLS-K experts)	
	COBYLA	SLSQP
MOPTA 30D	82(31+51)	68(31+37)

Table 7. MOPTA 30D - Number of calls (worst value of the 3 runs) of the black box function to reach the optimum (DOE samples + number of iterations) with SEGOMOE approach (KPLS or KPLS-K are the local experts selected for the MOE).

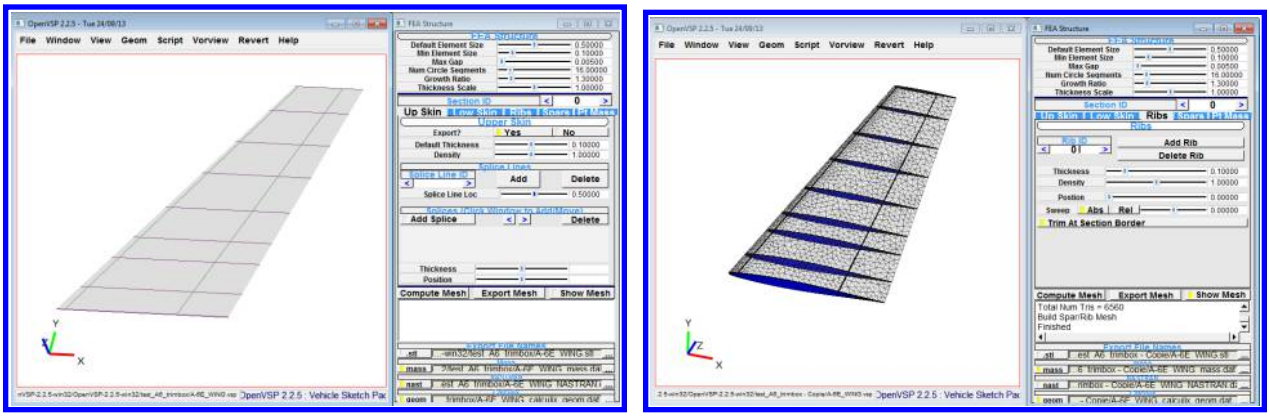
The MOPTA test case with increased numbers of variables has to be considered and optimisation problems with 50 and 124 dimensions (with the other design variables set to the known optimum) are under evaluation.

C. First results on wing design optimization

The previous test cases enable to assess the efficiency of SEGOMOE and its capacity to successfully handle problems with a few tens of design variables. The next step is then to confront this optimisation process to more realistic test cases, ie. optimisation problems relying on high fidelity tools such as CFD or CSM. First, a classical optimisation test case is tackled, dedicated to an aircraft wing design. A mono-disciplinary structural optimisation is set up with an aim of minimizing the weight of the internal structure. An extension to bi-disciplinary optimisation test case, including aerodynamic, is also introduced. This test case is built on open source codes for geometry modeler, structural and aerodynamic solvers (resp. OpenVSP, Calculix and VSPaero) which are described in the next section.

1. *Description of tools*

NASA open-source Vehicle Sketch Pad (VSP),[48] is a geometry modeling tool for conceptual aircraft design. This software facilitates the modeling of aircraft configurations without expending the expertise required for traditional Computer Aided Design (CAD) packages. Therefore Vehicle Sketch Pad (VSP) is able to handle rapid evaluation of advanced design concepts. A useful parametric geometry tool must not only depict the geometry, but it must translate the familiar description of an aircraft into a model which can be useful for engineering purposes such as CFD or FEA analysis by providing aerodynamic and structural meshing. OpenVSP "wing structure" sub-module can create the basic wing internal structure adding ribs, spars, skin and theirs thickness and generate a FEM meshing. The design is simple: spars and ribs are added and located for each section of the wing relatively to the wing geometry, maintaining parametric design. They are defined with a reference number starting at zero and located with a normalized position relative to chord length and/or wing span. Thickness of different components is then set up (default value 0.1) and a mesh can be asked to be generated. Figure 10 provides an overview of the OpenVSP GUI used to design the wing internal structure and mesh.



(a) Wing structure definition in OpenVSP

(b) Wing meshing in OpenVSP

Figure 10. Overview of OpenVSP wing structure sub-module.

CalculiX[49] is an open source software for Finite Element Method (FEM) solving, composed by two sub-softwares CalculiX CrunchiX (CCX) as a solver and CalculiX GraphiX (CGX) as a pre and post processing tool with a graphical interface. Both can be run in a batch mode.

CCX input file is the `<jobname>.inp` file which contains all the specified keys, essential for the solver. Creating an input file is quite easy as it just requires to make calls to external files such as meshing definition or load file, contains material properties, and uses keys for the solver outputs (e.g stress, strain, displacement,...). Therefore, as the geometry will not be created with the graphical interface, a few steps were required to correctly define the structure for the FEM solver and to show up parameters in order to implement the tool within the OpenMDAO framework.

The link between OpenVSP and CalculiX has already been studied and implemented by the University of Texas through VSP SAM[50,51] executable. VSP SAM stands for Structural Analysis Module and is a java script with a user-friendly GUI that sets up the CCX input file and manages a mass optimization. A similar approach was followed to implement OpenVSP and CalculiX. The process was entirely written in Python scripts that allow an easy connection between the different tools: the main steps of the implementation are identified on figure 11. Moreover, some specific developments have been made in order to increase the fidelity of the wing structure such as the definition of panels as sections or the implementation of local buckling factors.

VSPAERO[52] is a fast, linear, vortex lattice solver which also integrates actuator disks for aero-propulsive analysis. VSPAERO is closely linked to OpenVSP which produces an adequate degenerated geometry. This geometry is then used to calculate the pressure distribution, and thus forces, such as drag and lift. The skin friction drag of each component in a model can also be provided, through the use of flat-plate drag model. This solver will be used to provide the loads to the structural solver.

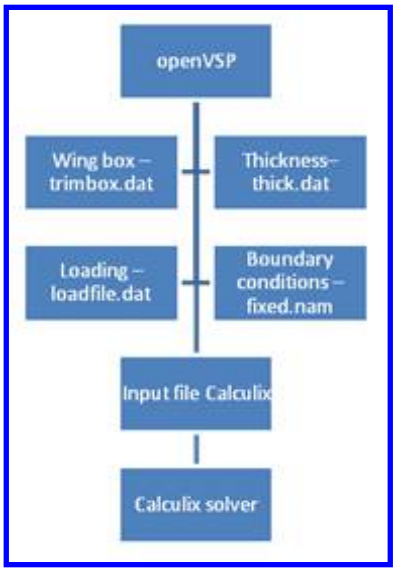


Figure 11. OpenVSP to Calculix process.

2. *Wing design optimization*

The implementation of CalculiX in the OpenMDAO framework requires to set this module as a Component, allowing OpenMDAO to recognize it as an object containing inputs and outputs. The CalculiX component is included in a Group and a function is implemented around it to use it as a black box. The entries of this black box, which are the design variables of our optimization problem, are the thicknesses of the 8 ribs, the 2 spars and the upper and lower skins. The wing's structure is subjected to a set of loads which have previously been computed by VSPaero.

The outputs are the wing mass which is our objective function, the deflections and twists of each section, the global and local buckling factors, and the stresses of the ribs, spars and skins. However, to simplify the problem, the outputs on which we are going to impose some constraints in our optimization problem are the spars, ribs and skins stresses.

The optimization problem is the following:

$$\begin{cases} \min \text{Wing mass} \\ w.r.t. \\ \text{spars thickness } \in [0.2, 0.4] \\ \text{skins thickness } \in [0.08, 0.15] \\ \text{ribs thickness } \in [0.045, 0.125] \\ s.t. \\ \max(\text{ spars stresses }) \leq 1.7\ 10^6 \\ \max(\text{ skins stresses }) \leq 1.7\ 10^6 \end{cases} \tag{16}$$

with a total of 12 design variables and 2 stress constraints.

This test case, mapped on figure 12 is a good opportunity to test SEGOMOE with COBYLA and SLSQP because it is relatively fast in both cases (about one hour to converge), it does not need too much calls to the objective function to approach the solution (about 30), and the solution point reaches some constraints in the design space while others are not active, thus allowing to test the behaviors of the methods facing both scenarii. The indicators of performance to assess the algorithms performances are their accuracy, convergence time and robustness with respect to the initial DOE. Because of the Latin Hypercube Sampling which produces different results when applied several times to the same problem and the initial point chosen by COBYLA which can vary, it is necessary to evaluate the mean values and dispersion over these indicators. For this purpose, table 8 compiles the results of about 50 optimizations, testing each algorithm with 3 different sizes of DOE and performing 8 runs for each of these choices. As the size of the design space is equal to 12, we consider 5, 13 and 24 sampling points for the initial DOE.

The table 8 demonstrates that both algorithms are quite sensitive to the intial DOE. The obtained results tend to prove that SLSQP needs less evaluations than COBYLA to converge for all initial DOE sizes.

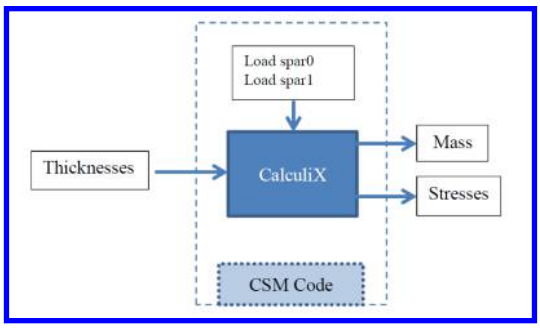


Figure 12. Mono-disciplinary optimization.

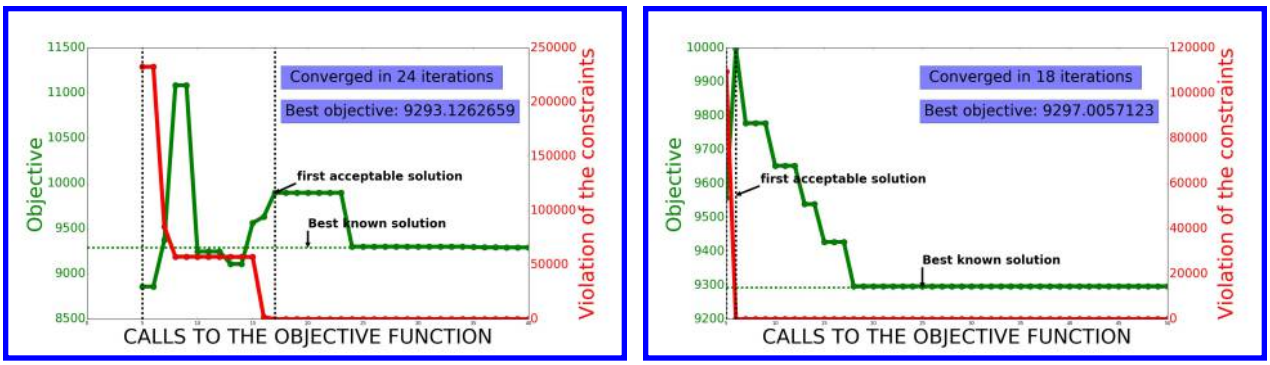
	COBYLA			SLSQP			SLSQP
Size of the initial DOE	5	13	24	5	13	24	direct
Relative Error							
Mean	4.91E-06	4.66E-04	1.21E-09	2.02E-03	1.55E-03	2.53E-03	5.7E-04
Standard deviation	1.18E-05	1.32E-03	1.42E-10	1.69E-03	1.38E-03	1.47E-03	(1 run)
Number of Evaluations							
Mean	19.4	20.7	27.2	16.6	18.8	26.3	250 iter
Standard deviation	4.44	5.03	1.47	2.24	4.78	1.25	(1 run)

Table 8. Mean values and standard deviations of the relative error between the best found solution and the best known solution (in %) and the number of evaluations required to converge. For each case, a total of 8 runs were used to compute these values. The convergence criterion is satisfied when the relative error is smaller than 0.5%, but the relative error in the table is the one after exactly 40 iterations. Note that these values compile the results of the runs which converged before the 40^{th} iteration. The results for a direct optimization with SLSQP are those of one run, which took 250 iterations to converge around the best known solution.

Nevertheless this behaviour is closely linked to the selected convergence criteria. Actually, at convergence, the SLSQP solution always exhibits a mean relative error higher than for COBYLA (up to 6 orders of magnitude). Therefore, a smaller convergence criterion would lead to a better performance of COBYLA algorithm. Another interesting point, is that, in both cases, a small initial DOE is sufficient to achieve a converged solution at minimal evaluations cost. In fact, starting from a relatively big DOE will lead to a quicker convergence but at a higher overall evaluation cost (due to the initial database). The results are compared with a direct optimization where SLSQP is launched on CalculiX without surrogate model. In this case, 250 iterations are required to converge.

Figure 13 shows typical convergence histories of the problem with SEGOMOE COBYLA and SEGOMOE SLSQP. The first part of the plots illustrates the search for a point that does not violate the constraints which is complex as only a subdomain of the design space verifies these properties. During this phase, the violation of the constraints can only decrease as this is the only criterion defining the consecutive best points. Then, as soon as a feasible solution is found, the criterion for the consecutive best points is to be an acceptable point with a lower objective function. During this phase, the violation of the constraints is zero and the objective function can only decrease.

The figure 14 shows that after an initial DOE (same size), SLSQP seems to explore more the design space than COBYLA. COBYLA tends to focus around the global optimum and find a better optimum than SLSQP. COBYLA or SLSQP are here compared using a multistart of 10 different points. With COBYLA, most of these optimizations focus around the same point (which happens to be the best solution) while SLSQP gets stuck in different areas. Nevertheless, SLSQP leads to a greater space exploration, but it would be wrong to conclude that SLSQP is better for finding global optima, as the phenomenon we observe here is more a downside of SLSQP than a strength.



(a) COBYLA (b) SLSQP

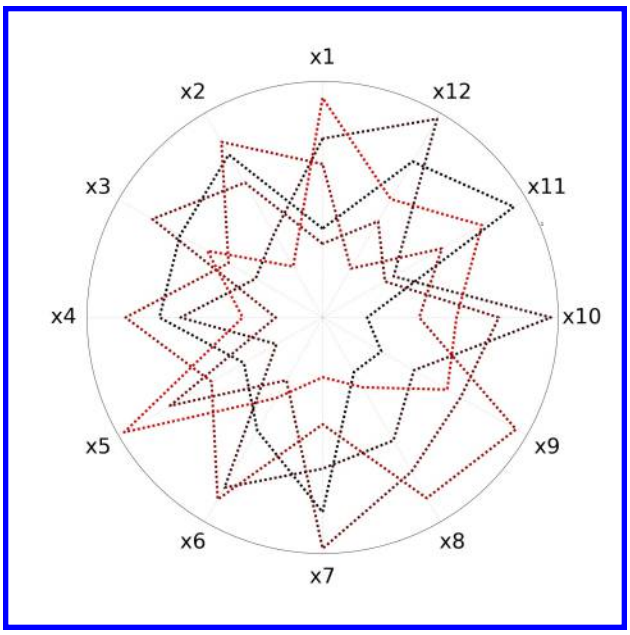
Figure 13. Example of convergence history with the two algorithms for SEGOMOE, after a DOE of 5 points. The consecutive best points are displayed in green, the sum of the constraints violations is displayed in red (0 as soon as a valid solution is found). Before the first acceptable solution is found, the best consecutive points are the one for which the sum of the constraints violation is smaller than the previous one.

3. Extension to aero-structural optimization

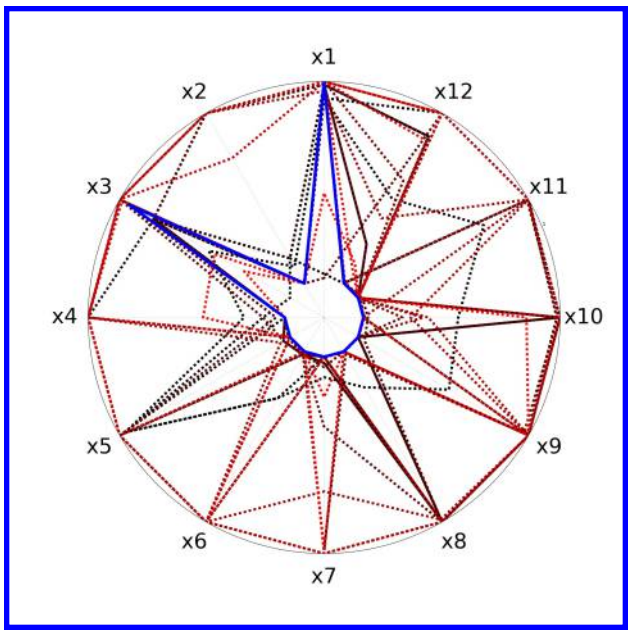
The next challenge is to couple an aerodynamic solver (VSPaero) with a structural solver (CalculiX) in order to reach the aero-elastic equilibrium. VSPaero is implemented as a Component in OpenMDAO in order to expose inputs and outputs. For any flight case, a fixed point iterator solver is set up in order to connect the wing deformation with the aerodynamic load and iterate on these values until the tolerance threshold between target values and ingoing ones is reached. In order to perform an aero-structural optimisation of the wing, the proposed approach is to create a package containing n-flight cases (or load cases), corresponding to n different fixed point iterator modules, possibly running in parallel, becoming a component for SEGOMOE method as illustrated on figure 15. Variables are still the material thicknesses and an MOE is created with a single mass as an output (the mass being only dependent of thicknesses, it will be identical for all flight case) and all constraints from the different flight cases as others outputs. In this way, a mass optimization is being set up corresponding to any structural sizing flight case authorizing less violating constraints and giving the minimum mass. This approach is under implementation within OpenMDAO framework and will provide a more realistic test case to assess the SEGOMOE algorithm.

VI. Conclusion

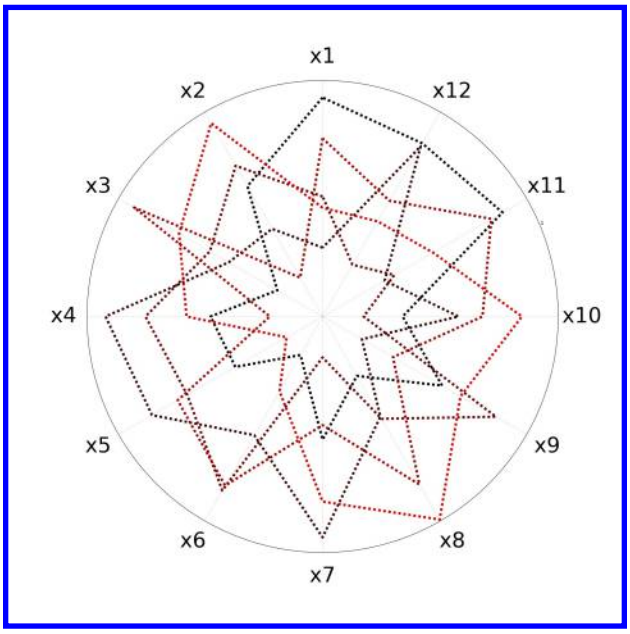
For a new aircraft configurations development, we need to integrate more accurate data coming from high fidelity analyses earlier in the design process. New methods have been investigated to obtain optimized configuration at reasonable computational cost. This paper presented a solution to tackle this kind of optimization process of complex design problem, through the use of enrichment strategy approach. The paper focused on the improvements made on: Mixture of Experts (kriging-based surrogates) and Enrichment process for optimization under constraints at high dimensions. SEGOMOE is a very promising algorithm for expensive black-box optimization under constraints in high-dimensions that extend the applicability of kriging-based methods. Based on the kriging properties, we are able to predict both the output variable (mean) and an estimate of the variance. The Efficient Global Optimization algorithm relies on an Expected Improvement criterion accounting for the exploration-exploitation trade-off. Maximizing the expected improvement balances exploration and exploitation because the expected improvement can be high because of large uncertainty in sparsely sampled region, or it can be high because of low kriging predictions. EGO was developed with kriging, even though it is applicable to any surrogate with a variance estimation. In order to handle high-dimensional design optimization problems with a large number of constraints, some improvements based on SEGO have been proposed in this paper. They rely on some combination of kriging-based models well adapted for high-dimensions (KPLS and KPLS-K models). These mixture of kriging-based models are used in an adaptive process SEGOMOE to select infill sample points in areas close to a minimum point. The final step was to implement this process within the OpenMDAO framework. The analytic and industrial test cases confirm that the proposed SEGOMOE algorithm was successful to minimize the



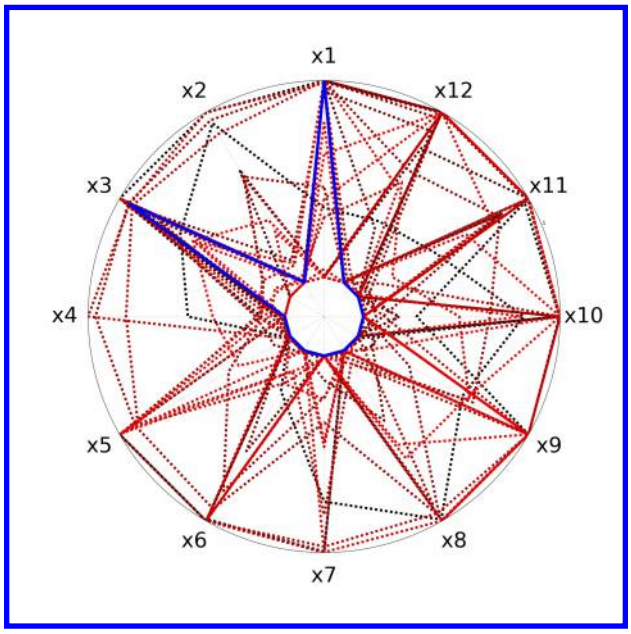
(a) Initial DOE with 5 points for COBYLA



(b) 35 enrichment Points with COBYLA. The chronological order of the calculations is given by plotting the points from black to red, and the best known solution in blue.



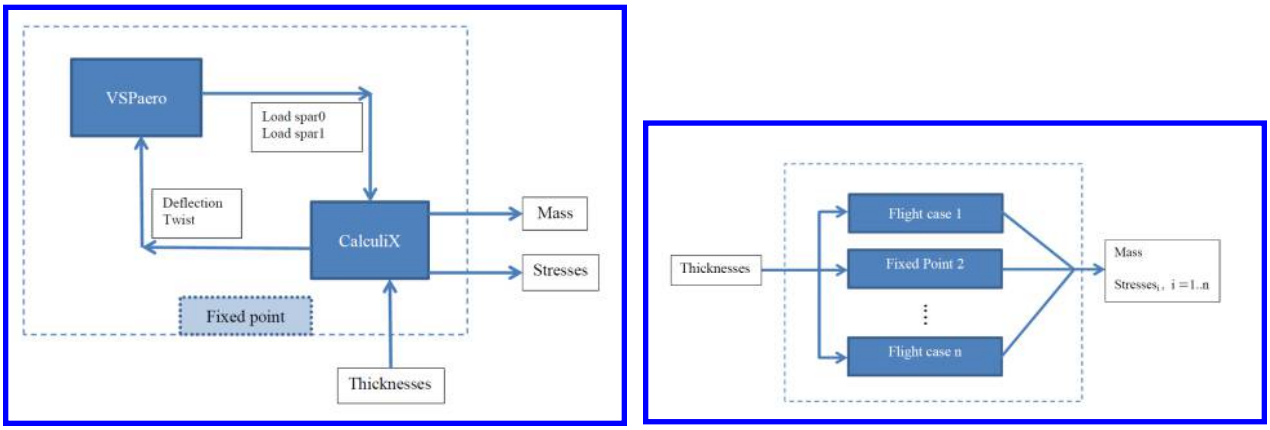
(c) Initial DOE with 5 points for SLSQP



(d) 35 enrichment Points with SLSQP. The chronological order of the calculations is given by plotting the points from black to red, and the best known solution in blue.

Figure 14. Radar chart for displaying domain exploration with COBYLA (top) or SLSQP (bottom). Each direction of the chart represents one of the 12 design variables, with the inner and outer radii limiting the design space.

number of black box evaluations. Nevertheless, the SEGOMOE still have to be confronted to more complex optimization problems, either through the increase of variables space dimension or to more realistic aircraft design problems. In addition some improvements are already foreseen on some steps of the algorithm. First, the enrichment process could be improved by changing the optimizers in the search of the maximum of the EI or WB2 criteria. Indeed, we could also use some genetic algorithm optimizers such as NBSGAII[31] or



(a) Fixed point (b) Objective function

Figure 15. The next step towards a more industrial-like problem. This model would capture the subtilities of the aero-structural trade offs that the aircraft designer needs to resolve, while taking care of creating a solution that suits the aircraft's mission as a whole. Each flight case has the structure of the fixed point presented here.

other gradient based optimizers like SNOPT[53] developed at Standford University. Another point could be investigation about the enrichment criterion such as the Stepwise Uncertainty Reliability strategy proposed in 54 for constrained global optimization. In parallel, as this process tends to enrich in areas where the number of samples is already high, the covariance matrix of the kriging model could become singular and so could not be inverted. Some algebraic techniques are already investigating to handle this limit. None of these drawbacks seems to be a limit to consider more complex design problems in aeronautical engineering.

Appendix

A. SEGOMOE algorithm

Algorithm 1 SuperEGO with Mixture Of Experts (SEGOMOE)

Input: Initial design of experiments (DOE) or initial database of design points $\mathbf{x}^{(0)}$
Output: Optimal variables $\mathbf{x}^*$, objective function f^*, and constraint values $c_i^\star \ \forall i = 1 \dots m$
0: Initiate design of experiments (DOE) to generate design points
for each DOE point **do**
  for each discipline i **do**
    1: Evaluate discipline analysis
    2: Compute discipline i output $y_i \ \forall i$
  end for
end for
for each discipline i **do**
  3: Build discipline surrogate model (MOE_i)
end for
4: Initiate enrichment optimization iteration
repeat
  for each discipline i **do**
    5: Evaluate discipline surrogate model i at the current point $\mathbf{x}$
    6: Compute metamodel mean $\hat{y_i}$ and variance $\hat{s_i}$
  end for
  7: Recombination of the metamodels $\hat{y_i}$ to compute approximations of the objective $\hat{f}$ and constraint functions $\hat{c_i}$ of the physical optimization problem (1)
  8: Compute mean and variance for the objective $(\hat{f}, \hat{s})$ and all constraints $(\hat{c_i}, \hat{s_i}) \ \forall i = 1 \dots m$, determine the $f_{\min}$ in the latest database
  9: Evaluate the enrichment criteria (EI or WB2) and the associated constraint functions of the SEGOMOE problem
  10: Compute objective and contraints of the SEGOMOE problem
until $10 \rightarrow 4$: enrichment optimization has converged
11: Initiate main SEGOMOE iteration
repeat
  for each discipline i **do**
    12: Evaluate discipline analysis at the newest design point
    13: Compute discipline i output y_i
    14: Update discipline surrogate model with newest design point
  end for
  15: Initiate enrichment optimization iteration
  repeat
    for each discipline i **do**
      16: Evaluate discipline surrogate model i at the current point $\mathbf{x}$
      17: Compute metamodel mean $\hat{y_i}$ and variance $\hat{s_i}$
    end for
    18: Recombination of the metamodels $\hat{y_i}$ to compute approximations of the objective $\hat{f}$ and constraint functions $\hat{c_i}$ of the physical optimization problem (1) at the newest design point
    19: Compute mean and variance for the objective $(\hat{f}, \hat{s})$ and all constraints $(\hat{c_i}, \hat{s_i}) \ \forall i = 1 \dots m$, determine the $f_{\min}$ in the latest database
    20: Evaluate the enrichment criteria (EI or WB2) and the associated constraint functions of the SEGOMOE problem
    21: Compute objective and contraints of the SEGOMOE problem
  until $21 \rightarrow 15$: enrichment optimization has converged
until $22 \rightarrow 11$: SEGOMOE optimization has converged

B. Definition of the academic test cases for constrained optimization

Let define first the *plog* transformation used in some of the constraints by

$$plog(x) = \begin{cases} log(1+x) & \text{if} \quad 0 \leq x \\ -log(1-x) & \text{if} \quad x \leq 0. \end{cases} \tag{17}$$

where *log* is the natural logarithm. This transformation is applied without changing the feasible region (for more details, please see.[55])

Pressure Vessel Design (PVD4)[56]

$$\begin{cases} \min 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3 \\ \texttt{s.t.} \\ g_1 = -x_1 + 0.0193x_3 \leq 0 \\ g_2 = -x_2 + 0.00954x_3 \leq 0 \\ g_3 = plog(-\pi x_3^2 x_4 - \frac{4}{3}\pi x_3^3 + 1296000) \leq 0 \\ 0 \leq x_1, x_2 \leq 1, \;\; 0 \leq x_3 \leq 50, \;\; 0 \leq x_4 \leq 240. \end{cases} \tag{18}$$

g_{02}[57]

$$\begin{cases} \min -\left| \frac{\sum_{i=1}^{d} \cos^4(x_i) - 2\prod_{i=1}^{d} \cos^2(x_i)}{\sqrt{\sum_{i=1}^{d} i x_i^2}} \right| \\ \texttt{s.t.} \\ g_1 = \frac{plog(-\prod_{i=1}^{d} x_i + 0.75)}{plog(10^d)} \leq 0 \\ g_2 = \frac{\sum_{i=1}^{d} x_i - 7.5d}{2.5d} \leq 0 \\ 0 \leq x_i \leq 10, \; i = 1, \cdots, d. \end{cases} \tag{19}$$

g_{07}[57]

$$\begin{cases} \min x_1^2 + x_2^2 + x_1x_2 - 14x_1 - 16x_2 + (x_3-10)^2 + 4(x_4-5)^2 + (x_5-3)^2 \\ \quad +2(x_6-1)^2 + 5x_7^2 + 7(x_8-11)^2 + 2(x_9-10)^2 + (x_{10}-7)^2 + 45 \\ \texttt{s.t.} \\ g_1 = \frac{4x_1+5x_2-3x_7+9x_8-105}{105} \leq 0 \\ g_2 = \frac{10x_1-8x_2-17x_7+2x_8}{370} \leq 0 \\ g_3 = \frac{-8x_1+2x_2+5x_9-2x_{10}-12}{158} \leq 0 \\ g_4 = \frac{3(x_1-2)^2+4(x_2-3)^2+2x_3^2-7x_4-120}{1258} \leq 0 \\ g_5 = \frac{5x_1^2+8x_2+(x_3-6)^2-2x_4-40}{816} \leq 0 \\ g_6 = \frac{0.5(x_1-8)^2+2(x_2-4)^2+3x_5^2-x_6-30}{834} \leq 0 \\ g_7 = \frac{x_1^2+2(x_2-2)^2-2x_1x_2+14x_5-6x_6}{788} \leq 0 \\ g_8 = \frac{-3x_1+6x_2+12(x_9-8)^2-7x_{10}}{4048} \leq 0 \\ -10 \leq x_i \leq 10, \; i = 1, \cdots, 10. \end{cases} \tag{20}$$

g_{10}[57]

$$\begin{cases} \min x_1 + x_2 + x_3 \\ \texttt{s.c.} \\ g_1 = -1 + 0.0025(x_4 + x_6) \leq 0 \\ g_2 = -1 + 0.0025(-x_4 + x_5 + x_7) \leq 0 \\ g_3 = -1 + 0.01(-x_5 + x_8) \leq 0 \\ g_4 = plog(100x_1 - x_1x_6 + 833.33252x_4 - 83333.333) \leq 0 \\ g_5 = plog(x_2x_4 - x_2x_7 - 1250x_4 + 1250x_5) \leq 0 \\ g_6 = plog(x_3x_5 - x_3x_8 - 2500x_5 + 1250000) \leq 0 \\ 10^2 \leq x_1 \leq 10^4, \;\; 10^3 \leq x_2, x_3 \leq 10^4, \\ 10 \leq x_i \leq 10^3, \; i = 4, 5, 6, 7, 8. \end{cases} \tag{21}$$

VII. Acknowledgments

The authors would like to thank Dimitri Bettebghor for providing IMAGE,[6] the MOE Matlab Code, Rémi Vauclin for his work on DOE and multifidelity kriging,[37] Joaquim R.R.A. Martins and Rhea P. Liem for their support and their help during comparisons with their MOE code.[35]

References

[1]RTO Applied Vehicle Technology Panel (AVT) Workshop, *Towards multi-level, multi-fidelity, multi-disciplinary optimization at ONERA*, May 2011.

[2]Carrier, G., Atinault, O., Dequand, S., Hantrais-Gervois, J., Liauzun, C., Paluch, B., Rodde, A., and Toussaint, C., "Investigation of a strut-braced wing configuration for future commercial transport," *28th Congress of the International Council of the Aeronautical Sciences (Brisbane, Australia)*, 2012.

[3]Jones, D. R., Schonlau, M., and Welch, W. J., "Efficient global optimization of expensive black-box functions," *Journal of Global optimization*, Vol. 13, No. 4, 1998, pp. 455–492.

[4]Haftka, R. T., Villanueva, D., and Chaudhuri, A., "Parallel surrogate-assisted global optimization with expensive functions – a survey," *Structural and Multidisciplinary Optimization*, 2016, pp. 1–11.

[5]Sasena, M., *Flexibility and efficiency enhancements for constrained global design optimization with Kriging approximations*, Ph.D. thesis, University of Michigan, 2002.

[6]Bettebghor, D., Bartoli, N., Grihon, S., Morlier, J., and Samuelides, M., "Surrogate modeling approximation using a mixture of experts based on EM joint estimation," *Structural and Multidisciplinary Optimization*, Vol. 43, No. 2, 2011, pp. 243–259, 10.1007/s00158-010-0554-2.

[7]Matheron, G., "Principles of geostatistics," *Economic geology*, Vol. 58, No. 8, 1963, pp. 1246–1266.

[8]Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P., "Design and analysis of computer experiments," *Statistical science*, 1989, pp. 409–423.

[9]Forrester, A. I., Sóbester, A., and Keane, A. J., "Multi-fidelity optimization via surrogate modelling," *Proceedings of the royal society A: mathematical, physical and engineering science*, Vol. 463, No. 2088, 2007, pp. 3251–3269.

[10]Bouhlel, M. A., Bartoli, N., Otsmane, A., and Morlier, J., "Improving kriging surrogates of high-dimensional design models by Partial Least Squares dimension reduction," *Structural and Multidisciplinary Optimization*, Vol. 53, No. 5, 2016, pp. 935–952.

[11]Wold, H., "Soft modeling by latent variables: the nonlinear iterative partial least squares approach," *Perspectives in probability and statistics, papers in honour of MS Bartlett*, 1975, pp. 520–540.

[12]Bouhlel, M. A., Bartoli, N., Otsmane, A., and Morlier, J., "Kriging and principal components for simulation with many inputs," *Mathematical Problems in Engineering*, 2016, accepted for publication.

[13]Powell, M. J., "A direct search optimization method that models the objective and constraint functions by linear interpolation," *Advances in optimization and numerical analysis*, Springer, 1994, pp. 51–67.

[14]Anjos, M. F. and Jones, D. R., "MOPTA 2008 Benchmark," 2009, http://www.miguelanjos.com/jones-benchmark.

[15]Mortished, C., Ollar, J., Jones, R., Benzie, P., Toropov, V., and Sienz, J., "Aircraft Wing Optimisation based on Computationally Efficient Gradient-Enhanced Kriging," *57th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference San Diego, California, USA*, 2016.

[16]Morris, M. D. and Mitchell, T. J., "Exploratory designs for computational experiments," *Journal of Statistical Planning and Inference*, Vol. 43, No. 3, 1995, pp. 381 – 402.

[17]Johnson, M. E., Moore, L. M., and Ylvisaker, D., "Minimax and maximin distance designs," *Journal of Statistical Planning and Inference*, Vol. 26, No. 2, Oct. 1990, pp. 131–148.

[18]Park, J.-S., "Optimal Latin-hypercube designs for computer experiments," *Journal of Statistical Planning and Inference*, Vol. 39, No. 1, April 1994, pp. 95–111.

[19]Jin, R., Chen, W., and Sudjianto, A., "An efficient algorithm for constructing optimal design of computer experiments," *Journal of Statistical Planning and Inference*, Vol. 134, No. 1, 2005, pp. 268–287.

[20]Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P., "Optimization by simulated annealing," *SCIENCE*, Vol. 220, No. 4598, 1983, pp. 671–680.

[21]Rasmussen, C. E., "Gaussian processes for machine learning," 2006.

[22]Krige, D. G., *A statistical approach to some mine evaluations and allied problems at the witwatersrand*, Master's thesis, University of Witwatersrand, 1951.

[23]Sacks, J., Schiller, S., and Welch, W., "Design for computer experiments," *Technometrics*, Vol. 31, No. 1, 1989, pp. 41–47.

[24]Welch, W. J., Buck, R. J., Sacks, J., Wynn, H. P., Mitchell, T. J., and Morris, M. D., "Screening, Predicting, and Computer Experiments," *Technometrics*, Vol. 34, No. 1, 1992, pp. 15–25.

[25]Forrester, A., Sobester, A., and Keane, A., *Engineering design via surrogate modelling: a practical guide*, John Wiley & Sons, 2008.

[26]Kleijnen, J. P., *Design and analysis of simulation experiments*, Vol. 230, Springer, 2015.

[27]Jones, E., Oliphant, T., Peterson, P., et al., "SciPy: Open source scientific tools for Python," 2001–, [Online; accessed 2016-04-27].

[28]Perez, R. E., Jansen, P. W., and Martins, J. R. R. A., "pyOpt: A Python-Based Object-Oriented Framework for Nonlinear Constrained Optimization," *Structures and Multidisciplinary Optimization*, Vol. 45, No. 1, 2012, pp. 101–118.

[29]Kenway, G. K. W., "pyOptSparse - PYthon OPTimization (Sparse) Framework," 2014, https://bitbucket.org/mdolab/pyoptsparse.

[30]Kraft, D. et al., *A software package for sequential quadratic programming*, DFVLR Obersfaffeuhofen, Germany, 1988.

[31]Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T., "A fast and elitist multiobjective genetic algorithm: NSGA-II," *Evolutionary Computation, IEEE Transactions on*, Vol. 6, No. 2, 2002, pp. 182–197.

[32]Hastie, T., Tibshirani, R., Friedman, J., and Franklin, J., "The elements of statistical learning: data mining, inference and prediction," *The Mathematical Intelligencer*, Vol. 27, No. 2, 2005, pp. 83–85.

[33]Jordan, M. I. and Jacobs, R. A., "Hierarchical mixtures of experts and the EM algorithm," *Neural computation*, Vol. 6, No. 2, 1994, pp. 181–214.

[34]Bettebghor, D. and Bartoli, N., "Approximation of the critical buckling factor for composite panels," *Structural and Multidisciplinary Optimization*, Vol. 46, No. 4, 2012, pp. 561–584.

[35]Liem, R. P., Mader, C. A., and Martins, J. R. R. A., "Surrogate Models and Mixtures of Experts in Aerodynamic Performance Prediction for Mission Analysis," *Aerospace Science and Technology*, Vol. 43, 2015, pp. 126–151.

[36]Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E., "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, Vol. 12, 2011, pp. 2825–2830.

[37]Vauclin, R., "Développement de modèles réduits multi-fidélité en vue de l'optimisation de structures aéronautiques," Tech. rep., ISAE-SUPAERO, July 2014.

[38]Le Gratiet, L. and Garnier, J., "Recursive co-kriging model for Design of Computer experiments with multiple levels of fidelity," *International Journal for Uncertainty Quantification*, 2014, pp. 365–386.

[39]Bouhlel, M.-A., *Optimisation auto-adaptative en environnement d'analyse multi-disciplinaire via les modèles de Krigeage combinés à la méthode PLS*, Ph.D. thesis, ISAE-SUPAERO, 2016, https://hal.archives-ouvertes.fr/tel-01293319.

[40]Lambe, A. B. and Martins, J. R., "Extensions to the design structure matrix for the description of multidisciplinary design, analysis, and optimization processes," *Structural and Multidisciplinary Optimization*, Vol. 46, No. 2, 2012, pp. 273–284.

[41]Viana, F. A., Haftka, R. T., and Watson, L. T., "Why not run the efficient global optimization algorithm with multiple surrogates," *Proceedings of the 51th AIAA/ASME/ASCE/AHS/ASC structures, structural dynamics, and materials conference, AIAA, Orlando, FL, USA. AIAA–2010–3090*, 2010.

[42]Heath, C. M. and Gray, J. S., "OpenMDAO: Framework for Flexible Multidisciplinary Design, Analysis and Optimization Methods," *8th AIAA Multidisciplinary Design Optimization Specialist Conference (MDO)*, Honolulu, Hawaii, 2012, pp. 1–13.

[43]Gray, J., Moore, K. T., Hearn, T. A., and Naylor, B. A., "Standard Platform for Benchmarking Multidisciplinary Design Analysis and Optimization Architectures," *AIAA Journal*, Vol. 51, No. 10, Oct 2013, pp. 2380–2394.

[44]Martins, J. R. R. A. and Hwang, J. T., "Review and Unification of Methods for Computing Derivatives of Multidisciplinary Computational Models," *AIAA Journal*, Vol. 51, 2013, pp. 2582–2599.

[45]Regis, R. G., "Constrained optimization by radial basis function interpolation for high-dimensional expensive black-box problems with infeasible initial points," *Engineering Optimization*, Vol. 46, No. 2, 2014, pp. 218–243.

[46]Jones, D., "Large-scale multi-disciplinary mass optimization in the auto industry," *MOPTA 2008 Conference (20 August 2008)*, 2008.

[47]Currie, J. and Wilson, D. I., "OPTI: lowering the barrier between open source optimizers and the industrial MATLAB user," *Foundations of computer-aided process operations, Savannah, Georgia, USA*, 2012, pp. 8–11.

[48]NASA Open Government Plan, "OpenVSP," http://www.openvsp.org/.

[49]Dhondt, G. and Wittig, K., "CalculiX," http://www.calculix.de/.

[50]Chaput, A. J. and Rizo-Patron, S., "Vehicle Sketch Pad Structural Analysis Module Enhancements for Wing Design," *Proceedings of the 50th AIAA Aerospace Sciences Meeting, Nashville, TN*, 2012.

[51]Aerospace Engineering and Engineering Mechanics, University of Texas at Austin, "VSP SAM, Vehicle Sketch Pad Structure Analysis Module," http://http://vspsam.ae.utexas.edu/.

[52]NASA Open Government Plan, "VSPaero," http://www.openvsp.org/wiki/doku.php?id=vspaerotutorial.

[53]Gill, P. E., Murray, W., and Saunders, M. A., "SNOPT: An SQP algorithm for large-scale constrained optimization," *SIAM review*, Vol. 47, No. 1, 2005, pp. 99–131.

[54]Picheny, V., "A Stepwise uncertainty reduction approach to constrained global optimization." *AISTATS*, 2014, pp. 787–795.

[55]Regis, R. G. and Shoemaker, C. A., "A quasi-multistart framework for global optimization of expensive functions using response surface models," *Journal of Global Optimization*, Vol. 56, No. 4, 2013, pp. 1719–1753.

[56]Carlos, A. C. and Efrén, M., "Constraint-handling in genetic algorithms through the use of dominance-based tournament selection," *Advanced Engineering Informatics*, Vol. 16, 2002, pp. 2002.

[57]Michalewicz, Z. and Schoenauer, M., "Evolutionary Algorithms for Constrained Parameter Optimization Problems," *Evolutionary Computation*, Vol. 4, 1996, pp. 1–32.