A Comparative Evaluation of Surrogate Models for Transonic Wing Shape Optimization

Emiliano Iuliano

Abstract The paper details a comparative analysis of different models able to provide a fast response within a surrogate-based shape optimization process. Kriging, Radial Basis Function Network (RBFN) and Proper Orthogonal Decomposition in combination with RBFNs (POD+RBFN) are employed as fitness function evaluators within the framework of evolutionary algorithms (EAs). The surrogate-assisted optimization consists of initializing the surrogate with space-filling samples, improving the accuracy by adding a series of "smart" samples through specifically designed in-fill criteria and finally optimizing on the surrogate. The test case is represented by the large scale shape optimization of a transonic wing in viscous flow and in multi-design point conditions. Optimization results obtained with the surrogates by fixing the total computational budget are presented: this procedure allows to make a fair comparison between the models and their performance during the optimization process.

Introduction

In real–world engineering design applications, high-fidelity simulations and reliable answers in short time are essential and fundamental requirements. Of course, they are often conflicting, especially when fluid dynamics is among the physical disciplines to be solved: indeed, computational fluid dynamics (CFD) simulations of complex configurations are still time–consuming and, considering also the high number of CFD simulations required by global optimization approaches, this strongly hampers the usage of such methods in engineering design.

Surrogate-based optimization (SBO) may provide an interesting answer to this issue as it relies on a fast response model to be used during optimization while

E. Iuliano (✉)
Fluid Mechanics Department, Multidisciplinary Analysis and Optimization Group,
Centro Italiano Ricerche Aerospaziali (CIRA),
Via Maiorise, 81043 Capua, Italy
e-mail: e.iuliano@cira.it

© Springer International Publishing AG, part of Springer Nature 2019

E. Andrés-Pérez et al. (eds.), *Evolutionary and Deterministic Methods for Design Optimization and Control With Applications to Industrial and Societal Problems*, Computational Methods in Applied Sciences 49, https://doi.org/10.1007/978-3-319-89890-2_11

invoking the "truth" model (i.e., the CFD simulation) to confirm the choice made by the surrogate. Several researchers have focused their attention on such topic, both from a theoretical (Forrester and Keane 2009; Braconnier et al. 2011; Viana et al. 2012) and application (Robinson et al. 2006; Booker et al. 1999; Mack et al. 2007) point of view. As a consequence, a varied amount of methods exist which differ substantially for the choice of the surrogate model (e.g., type, single or multiple), the approach to build the surrogate (e.g., optimize the generalization error or likelihood functions), the strategy for updating and improving the surrogate (e.g., evaluate surrogate minimizers, use in-fill criteria, random choice) and the optimization method (e.g., type, global or local or both).

The present paper proposes different choices of the surrogate model to be used within a SBO cycle with different updating strategies. The problem at hand is the multi-point shape optimization of a wing in viscous transonic conditions: such a problem stems as a large-scale and real-world optimization as it involves several design parameters and black-box CFD-based functions. As a consequence, in principle it cannot be handled by whatever methodology and the main aim is to provide arguments in support of the successful usage of accurate "optimal" surrogates and global optimization techniques.

Surrogate Models

This section is devoted to introduce the mathematical basis of the meta-models which will be used for surrogate-based optimization. Kriging and Radial Basis Function Network models work with scalar information (e.g., the objective function values) and are able to predict the response function at each location of the design space. On the other hand, the Proper Orthogonal Decomposition is coupled to Radial Basis Function Network models to deal with vector quantities (e.g., the flow field) and, thus, to inject more physics information within the surrogate training process.

Kriging

The Kriging model is built on the assumption that the training data obey a Gaussian process with an assumed form for the mean function and the covariance between data points. A Kriging surrogate models the response of interest $f(\mathbf{x})$ as a realization of a regression model h and a stochastic process z (Martin and Simpson 2005):

$$f(\mathbf{x}) = h(\beta, \mathbf{x}) + z(\mathbf{x}) \tag{1}$$

$$h(\beta, \mathbf{x}) = \mathbf{h}\beta \tag{2}$$

$$\mathbb{E}[z(\mathbf{x_1}), z(\mathbf{x_2})] = \sigma_k^2 R(\theta, \mathbf{x_1}, \mathbf{x_2}) \tag{3}$$

where β are the regression coefficients and $\mathbf{h}$ is the regression vector. The stochastic process z is assumed to have zero mean, process variance σ_k^2 and covariance model $R(\theta, \mathbf{x_1}, \mathbf{x_2})$ between $z(\mathbf{x_1})$ and $z(\mathbf{x_2})$ with parameters vector θ. The covariance model between function values is assumed to be only a function of the distance between points. Given the training sites $\{\mathbf{x}_j\}_{j=1,\ldots,M}$, the covariance matrix is given by $K_{ij} = R(\theta, \mathbf{x}_i, \mathbf{x}_j)$. Multi-dimensional covariance is built up using a tensor product of one-dimensional covariance functions:

$$R(\theta, \mathbf{x}_i, \mathbf{x}_j) = \prod_p^D Kr\left(\left|\frac{x_{ip} - x_{jp}}{\theta_p}\right|\right)$$

where D is the dimension of the problem, θ_p is the length scale in the p-th dimension, x_{ip} is the p-th component of the vector $\mathbf{x}_i$ and Kr is the one-dimensional Matern function. The latter function is computed as:

$$Kr(d) = \exp\left(-\sqrt{2\nu}d\right)\frac{\Gamma(t+1)}{\Gamma(2t+1)}\sum_{i=0}^{t}\frac{(t+i)!}{i!(t-i)!}\left(\sqrt{8\nu}d\right)^{t-i}$$

with Γ the Gamma function, $\nu = t + 1/2$ and three possible values of the parameter t:

$$Kr(d) = \begin{cases} \exp(-d) & \text{for } t = 0 \\ \left(1 + \sqrt{3}d\right)\exp(-\sqrt{3}d) & \text{for } t = 1 \\ \left(1 + \sqrt{5}d + \frac{5}{3}d^2\right)\exp(-\sqrt{5}d) & \text{for } t = 2 \end{cases}$$

Noise terms can be added along the covariance matrix diagonal in order to improve the matrix conditioning and to obtain a regressive behavior when dealing with noisy functions. The covariance matrix becomes $K_{ij} = R(\theta, \mathbf{x}_i, \mathbf{x}_j) + \lambda\delta_{ij}$, where the Kronecker convention has been used and λ is the noise ratio. The response function can be estimated at a generic location $\mathbf{x}$ as

$$\hat{f}(\mathbf{x}) = \mathbf{H}\hat{\beta} + \mathbf{k}^T\mathbf{K}^{-1}(\mathbf{f} - \mathbf{H}\hat{\beta}) \tag{4}$$

where $\mathbf{H}$ is the matrix of linear equations constructed using the regression function and the training sites, $\hat{\beta}$ is the generalized least square estimate of β, $\mathbf{K}$ is the covariance matrix, $\mathbf{k}$ is the covariance vector between the generic design site $\mathbf{x}$ and the training sites, and $\mathbf{f} = [f_1, f_2, \ldots, f_M]^T$ is the vector of the M training data (which corresponds to the given training dataset). One of the main advantages of the Kriging model is that it provides also an estimate of the prediction variance:

$$\hat{s}^2(\mathbf{x}) = \hat{\sigma}_k^2\left[1 - \mathbf{k}^T\mathbf{K}^{-1}\mathbf{k} + \mathbf{u}^T\left(\mathbf{H}^T\mathbf{K}^{-1}\mathbf{H}\right)^{-1}\mathbf{u}\right] \tag{5}$$

where $\hat{\sigma}_k^2$ is the estimated process variance, $\mathbf{u} = \mathbf{H}^T\mathbf{K}^{-1}\mathbf{k} - \mathbf{h}$ and $\mathbf{h} = [h_1, h_2, \ldots, h_M]^T$. In the most general case, both the response prediction $\hat{f}$ and the prediction variance $\hat{s}^2$ are function of the so called hyperparameters, i.e. the length scales θ_p, the process variance $\hat{\sigma}_k^2$ and the noise magnitude λ. Two methods are here used to find the optimal values of the hyperparameters, hereinafter referred to as "Full" and "Partial". The optimization of the hyperparameters is performed by calling the NLopt library (available online at http://ab-initio.mit.edu/nlopt) and implementing a sequential global–local approach: first, the search space is globally explored by means of the evolutionary strategy ESCH (da Silva Santos et al. 2010); then, starting from the best solution of the ESCH algorithm, a local refinement is carried out with a reviewed version of the Nelder-Mead simplex algorithm (Richardson and Kuester 1973).

Full Optimization

This formulation determines the regression parameters based on an optimality condition and fits all other covariance parameters (length scales, process variance and noise level) through maximization of the likelihood function. The likelihood formula for a Gaussian process with a regression mean function is given by:

$$\log p(f|x;\theta_p) = -\frac{1}{2}\mathbf{f}^T\mathbf{K}^{-1}(\mathbf{f} - \mathbf{H}\hat{\beta}) - \frac{1}{2}\log|\mathbf{K}| - \frac{1}{2}\log|\mathbf{A}| - \frac{M-S}{2}\log 2\pi$$

where M is the number of training points, S is the number of terms in the regression and the regression matrix A is defined as:

$$\mathbf{A} = \mathbf{H}^T\mathbf{K}^{-1}\mathbf{H}$$

The optimal regression parameters are given by:

$$\hat{\beta} = \mathbf{A}^{-1}\mathbf{H}^T\mathbf{K}^{-1}\mathbf{f}$$

Partial Optimization

This formulation determines the process variance and regression parameters based on the optimality condition and only performs optimization over the covariance length scales θ_p. The likelihood formula reduces to:

$$\log p(f|x;\theta_p) = -\frac{M}{2}\log\hat{\sigma}_k^2 - \frac{1}{2}\log|\hat{\mathbf{K}}| - \frac{M}{2} - \frac{M}{2}\log 2\pi$$

where the optimal process variance has been estimated as:

$$\hat{\sigma}_k^2 = \frac{(\mathbf{f} - \mathbf{H}\hat{\beta})^T \hat{\mathbf{K}}^{-1} (\mathbf{f} - \mathbf{H}\hat{\beta})}{M}$$

and

$$\mathbf{K} = \hat{\sigma}_k^2 \hat{\mathbf{K}}$$

This final formula is a function of the length scales, θ_p, and the noise level ratio λ. The optimization is performed only over the length scales and the noise level ratio is fixed throughout the optimization. A typical choice is to set the noise level λ to a small fraction of the process variance $\hat{\sigma}_k$.

Radial Basis Function Network

A Radial Basis Function is a real valued function whose value depends on the Euclidean distance from a point called centre. A RBF network uses a linear combination of radial functions. A RBF model can be expressed as

$$f(\mathbf{x}, \theta_1, \ldots, \theta_M, \lambda) = \sum_{i=1}^{M} k_i(\lambda) r(|\mathbf{x} - \mathbf{x}_i|, \theta_i) \tag{6}$$

where the approximating function is represented by a sum of M RBFs r, each associated with a different center $\mathbf{x}_i$, weighted by real valued weights k_i (regularized through parameter λ) and characterized by width parameters θ_i. Hence, an RBF network can be defined as a weighted sum of translations of radially symmetric basis function. Typical RBFs kernel r used here are:

$$r(d, \theta) = \begin{cases} \exp\left(-\frac{d^2}{\theta^2}\right) & \text{Gaussian} \\ \sqrt{1 + \frac{d^2}{\theta^2}} & \text{Multi–quadric} \\ \frac{1}{\sqrt{1+\frac{d^2}{\theta^2}}} & \text{Inverse multi–quadric} \\ \left(\frac{d}{\theta}\right)^2 \ln \frac{d}{\theta} & \text{Thin plate spline} \\ 1 - 30\left(\frac{d}{\theta}\right)^2 - 10\left(\frac{d}{\theta}\right)^3 + & \\ +45\left(\frac{d}{\theta}\right)^4 - 6\left(\frac{d}{\theta}\right)^5 - 60\left(\frac{d}{\theta}\right)^3 \log\left(\frac{d}{\theta}\right) & \text{Wendland } C^2 \text{ thin plate spline} \end{cases}$$

Once decided the RBF kernel and supposing that the "optimal" width parameters have been already computed in some way, the RBF network is defined only by the weights k_i. They are made function of a regularization parameter λ (also known as ridge regression parameters in the RBF literature) to avoid overfitting and improve the interpolation matrix conditioning. Indeed, the weights can be found by imposing the interpolation condition (Fasshauer and Zhang 2007) on the training set which in turn results in solving the linear system:

$$\mathbf{Rk} = \mathbf{f} \tag{7}$$

where

$$\mathbf{R} = \begin{Bmatrix} r(0,\theta_1)+\lambda & \dots & r(|\mathbf{x}_1-\mathbf{x}_M|,\theta_M) \\ r(|\mathbf{x}_2-\mathbf{x}_1|,\theta_1) & \dots & r(|\mathbf{x}_2-\mathbf{x}_M|,\theta_M) \\ \vdots & \vdots & \vdots \\ r(|\mathbf{x}_M-\mathbf{x}_1|,\theta_1) & \dots & r(0,\theta_M)+\lambda \end{Bmatrix}$$

$\mathbf{k} = [k_1, k_2, \dots, k_M]^T$ are the RBF weights and $\mathbf{f} = [f_1, f_2, \dots, f_M]^T$ are the function values at the training points.

The width parameters have a significant influence both on the accuracy of the RBF model and on the conditioning of the solution matrix. In particular, it has been found (Gutmann 2001) that interpolation errors become high for very small and very large values of θ, while the condition number of the coefficient matrix increases with increasing values of θ. Therefore, they have to be "optimal" in the sense that a tuning of the width parameters is needed to find the right trade–off between interpolation errors and solution stability (Fasshauer and Zhang 2007). Generally speaking, two cases can be considered:

- identical scalar widths $\theta_i = \theta$ are used for all RBF kernels;
- different scalar width θ_i is used for each RBF kernel.

Here, the first option is chosen, therefore in the following a unique scalar width θ will be considered for each RBF center. An accurate RBF model is obtained by letting the algorithm autonomously choose the kernel function type *and* optimizing the width parameters. The algorithm is based on the Leave–One–Out cross–validation strategy to compute an error norm to be minimized; the procedure is similar to the one described in (Tenne and Armfield 2008) and is here outlined:

1. all the aforementioned kernel functions are used for training on the current training set;
2. the Leave–One–Out (LOO) error norm is considered as merit function to determine the best combination of RBF kernel and width parameter. The optimal RBF network is thus selected by choosing the width parameter which give the lowest LOO error norm, defined as:

$$\varepsilon_{LOO}(\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_M, \theta, \lambda) = \sqrt{\frac{1}{M}\sum_{j=1}^{M}[f_j - \hat{f}_{-j}(\mathbf{x}_j, \theta, \lambda)]^2}$$

   where f_j is the value of the function at the j^{th} training site $\mathbf{x}_j$ and $\hat{f}_{-j}$ is the RBF prediction at $\mathbf{x}_j$ when the model is trained without $\mathbf{x}_j$ and f_j. The computation of the M terms $\hat{f}_{-j}$ does not require to train M RBF models, indeed it can be computed effortlessly thanks to Rippa's formula (Rippa 1999);
3. for each kernel, the width parameter θ and regularization parameter λ are found by solving:

$$\min_{\theta,\lambda} \varepsilon_{LOO}(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M, \theta, \lambda) \tag{8}$$

The optimization is performed by using the same algorithms for searching the Kriging hyperparameters.

POD + Radial Basis Function Networks

The Proper Orthogonal Decomposition (POD) is used to extract the main features of a set of computed flow fields as a series of POD basis vectors with associated coefficients (Iuliano 2011; Iuliano and Quagliarella 2011). Given the three spatial coordinates (ξ, υ, ζ) of the computational mesh points and the general snapshot vector $\mathbf{s}$, let $\{\mathbf{x}_j\}$ be a set of design vectors (e.g., sampled from the design space with a DoE technique) and $\{\mathbf{s}_j\}$ the corresponding snapshot, i.e. column vectors containing the volume grid and flow variables as obtained from a CFD solution:

$$\begin{aligned}
\mathbf{s} &= (\mathbf{s}_{\text{grid}}, \mathbf{s}_{\text{flow}})^T \\
\mathbf{s}_{\text{grid}} &= (\xi_1, \ldots, \xi_q, \upsilon_1, \ldots, \upsilon_q, \zeta_1, \ldots, \zeta_q) \\
\mathbf{s}_{\text{flow}} &= (\rho_1, \ldots, \rho_q, \rho\xi'_1, \ldots, \rho\xi'_q, \rho\upsilon'_1, \ldots, \rho\upsilon'_q, \\
&\quad \rho\zeta'_1, \ldots, \rho\zeta'_q, p_1, \ldots, p_q)
\end{aligned}$$

where q is the number of mesh nodes involved in the POD computation, ρ is the flow density, $(\xi', \upsilon', \zeta')$ are the three Cartesian velocity components and p is the static pressure. The computational mesh has been included in the POD snapshot to let the SVD basis catch the coupling effects between space location and state field. Hence, once the surrogate model is built, not only a flow field can be computed, but also an approximation of the volume mesh. Such a surrogate model would be able to catch, although in a reduced order form, the cross effects of geometry modification and aerodynamic flow change. As the total number of variables is eight (three mesh variables and five flow variables), the global size of the snapshot is $N = 8 \times q$.

Starting from the vectors $\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_M$ obtained by CFD expensive computations for a representative set of design sites $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M$, finding a Proper Orthogonal Decomposition means to compute a linear basis of vectors to express any other $\mathbf{s}_j \in \mathbb{R}^N$ with the condition that this basis is optimal in some sense. To compute the optimal basis, we first define the snapshot deviation matrix

$$\mathbf{P} = \left(\mathbf{s_1} - \bar{\mathbf{s}}\ \mathbf{s_2} - \bar{\mathbf{s}} \cdots \mathbf{s_M} - \bar{\mathbf{s}}\right)$$

where the ensemble mean vector is computed as

$$\bar{\mathbf{s}} = \frac{1}{M} \sum_{j=1}^{M} \mathbf{s}_j$$

The POD decomposition is obtained by taking the singular value decomposition (SVD) of $\mathbf{P}$

$$\mathbf{P} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T = \mathbf{U} \begin{pmatrix} \sigma_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_M \\ 0 & \cdots & 0 \end{pmatrix} \mathbf{V}^T \tag{9}$$

with $\mathbf{U} \in \mathbb{R}^{N\times N}$, $\mathbf{V} \in \mathbb{R}^{M\times M}$, $\boldsymbol{\Sigma} \in \mathbb{R}^{N\times M}$ and the singular values $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_M \geq 0$. The POD basis vectors, also called POD modes, are the first M column vectors of the matrix $\mathbf{U}$, while the POD coefficients $\alpha_i(\mathbf{x}_j)$ are obtained by projecting the snapshots onto the POD modes:

$$\alpha_i(\mathbf{x}_j) = (\mathbf{s}_j - \bar{\mathbf{s}}, \boldsymbol{\phi}_i) \tag{10}$$

If a fluid dynamics problem is approximated with a suitable number of snapshots from which a rich set of basis vectors is available, the singular values become small rapidly and a small number of basis vectors are adequate to reconstruct and approximate the snapshots as they preserve the most significant ensemble energy contribution. In this way, POD provides an efficient mean of capturing the dominant features of a multi–degree of freedom system and representing it to the desired precision by using the relevant set of modes. The reduced order model is derived by projecting the CFD model onto a reduced space spanned by only some of the proper orthogonal modes or POD eigenfunctions. This process realizes a kind of lossy data compression through the following approximation

$$\mathbf{s}_j \simeq \bar{\mathbf{s}} + \sum_{i=1}^{\hat{M}} \alpha_i(\mathbf{x}_j)\boldsymbol{\phi}_i \tag{11}$$

where

$$\hat{M} \leq M \implies \sum_{i=1}^{\hat{M}} \sigma_i^2 \geq \varepsilon \sum_{i=1}^{M} \sigma_i^2 \tag{12}$$

and ε is a predefined energy level. In fact, the truncated singular values fulfils the relation

$$\sum_{i=\hat{M}+1}^{M} \sigma_i^2 = \varepsilon_{\hat{M}}$$

If the energy threshold is high, say over 99% of the total energy, then $\hat{M}$ modes are adequate to capture the principal features and approximately reconstruct the dataset. Thus, a reduced subspace is formed which is only spanned by $\hat{M}$ modes.

Equation 11 allows to get a POD approximation of any snapshot s_j belonging to the ensemble set. Indeed, the model does not provide an approximation of the state vector

at design sites which are not included in the original training dataset. In other words, the POD model by itself does not have a global predictive feature, i.e. over the whole design space. As the aim is to exactly reproduce the sample data used for training and to consistently catch the local data trends, a Radial Basis Function (RBF) network answers to these criteria and has been chosen as POD coefficients interpolation. The procedure to build optimal RBF models for POD modal coefficients is the same as described in Section "Radial Basis Function Network".

As a results, the pseudo–continuous prediction of the flow field at a generic design site **x** is then expressed as:

$$\mathbf{s}(\mathbf{x}) = \bar{\mathbf{s}} + \sum_{i=1}^{\hat{M}} \alpha_i(\mathbf{x})\boldsymbol{\phi}_i \tag{13}$$

This provides an accurate surrogate model which combines design of experiments for sampling, CFD for training, POD for model reduction and RBF network for global approximation. In conclusion, an explicit, global, low–order and physics–based model linking the design vector and the state vector has been derived and will be used as surrogate model. Examples of application and validation of the proposed POD/RBF surrogate model have been already provided in recent papers (Iuliano 2011; Iuliano and Quagliarella 2013).

Adaptive Sampling Strategy

Supposing that a surrogate model has been already trained, the training set is enriched by adding new samples, then the surrogate model is rebuilt and globally optimized. Hence, an iterative scheme is used for surrogate-based optimization: in the previous iteration, optimal candidates from the surrogate minimization are selected and passed to the next iteration; in the next iteration, the new samples are evaluated via the true, high-fidelity model and re-injected into the training set upon which the surrogate is updated. The aim of such an iterative scheme is to increase the quality and potential of the surrogates to be minimized, presumably driving to true optimality quickly. Of course, as this approach relies totally on the surrogate model and its prediction, it may drive the process towards local minima from which the surrogate model can no longer escape.

The weak point is considering the enrichment with new samples as a purely "exploitation" process and ignoring the "explorative" behaviour. Prior or during the optimization on the surrogate, we need to mix the knowledge from the available data, the surrogate prediction and an estimation of its predictive capability: we need to have a "smarter" selection of new points. However, the strategy for updating a surrogate model is heavily dependent on its type and scope and, in principle, has to be tailored on it. Indeed, the addition of new samples must follow some specific criteria that may be very different depending on the purpose of the training process.

For instance, Latin Hypercube Sampling has been designed to satisfy space-filling requirements and obtain a good coverage of the design space.

The present approach gives emphasis to the optimization process by proposing sampling strategies which are able to "adapt" to the response function. Most of the adaptive sampling approaches pursue the exploration/exploitation trade–off, where exploration means sampling away from available data, where the prediction error is supposedly high, while exploitation means trusting the model prediction, thus sampling where the surrogate provides global minima. It is clear that a trade-off between the two behaviors is needed: indeed, exploration is useful for global searching, but it may lead to unveil uninteresting regions of the design space; on the other hand, exploitation helps to improve the local accuracy around the predicted optima, but it may result in local minima entrapment.

Here, balanced explorative in-fill criteria are designed for a generic surrogate model and are formulated in terms of an auxiliary function which has to be maximized. The balanced criterion, hereinafter referred to as "EI-like", has been designed to mimic the same rationale of the Expected Improvement criterion, usually coupled to a Kriging-based surrogate in the well-known EGO algorithm by Jones (1998). The present approach, represents a generalization of that method as, for a generic surrogate model, the information about the uncertainty of the surrogate is not available, while a Kriging model, being a Gaussian process, provides an estimate of the prediction variance together with the prediction itself. The auxiliary function, also referred to as potential of improvement, is designed to have the same form of the Expected Improvement function.

Given $\mathbf{x}$ the generic design space location, $\hat{f}(\mathbf{x})$ the surrogate response, X_n the dataset of the training samples collected so far, F_{X_n} the corresponding values of the true objective function, f_{max} and f_{min} the maximum and minimum values in F_{X_n}, the potential of improvement function ("EI-like" function) is defined as follows:

$$v(\mathbf{x}, \hat{f}(\mathbf{x}), X_n, F_{X_n}) = [f_{min} - \hat{f}(\mathbf{x})]\Phi\Big[\frac{f_{min} - \hat{f}(\mathbf{x})}{\hat{s}(\mathbf{x})}\Big] + \hat{s}(\mathbf{x})\phi\Big[\frac{f_{min} - \hat{f}(\mathbf{x})}{\hat{s}(\mathbf{x})}\Big]$$

where $\hat{s}(\mathbf{x})$ is an estimate of the prediction error and $\Phi(\mathbf{x})$ and $\phi(x)$ are respectively the cumulative distribution and probability density functions of a standard normal distribution. The prediction error is estimated as follows:

$$\hat{s}(\mathbf{x}) = L(\mathbf{x})\frac{\min_{\mathbf{x}_i \in X_n} \|\mathbf{x} - \mathbf{x}_i\|_2}{\max_{\mathbf{x}_i, \mathbf{x}_j \in X_n} \|\mathbf{x}_i - \mathbf{x}_j\|_2} \exp\Big(-\gamma \frac{\max_{\mathbf{x}_i, \mathbf{x}_j \in X_n} \|\mathbf{x}_i - \mathbf{x}_j\|_2}{\min_{\mathbf{x}_i \in X_n} \|\mathbf{x} - \mathbf{x}_i\|_2}\Big)$$

where $L(\mathbf{x})$ is an estimate of the Lipschitz constant at $\mathbf{x}$ and γ is a tuning parameter. The Lipschitz constant is defined as:

Definition 1 Given a domain D and a function f defined in D, the Lipschitz constant is the smallest constant $L > 0$ in the Lipschitz condition, namely the non negative number:

$$L_{f,D} := \sup_{\substack{x_1,x_2 \in D \\ x_1 \neq x_2}} \frac{|f(x_1) - f(x_2)|}{|x_1 - x_2|}$$

The following algorithm has been designed to obtain an estimate of the Lipschitz constant at each training sample:

Algorithm 1 Lipschitz constant estimation

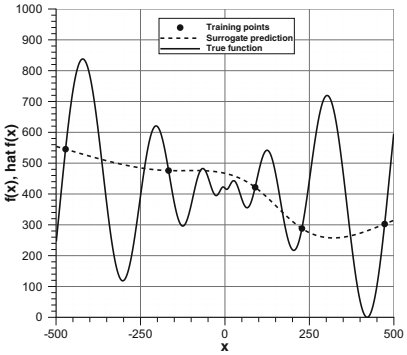
1: compute the K-means clusters $K_{j,j=1,r}$ of the set $X_n = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ with $r = int(\frac{n}{d})$
2: **for all** sample $\mathbf{x}_i \in X_n$ **do**
3: Say K_i the cluster containing $\mathbf{x}_i$
4: **for all** sample $\mathbf{x}_j \in K_i, \mathbf{x}_j \neq \mathbf{x}_i$ **do**
5: compute $L_{ij} = \frac{|f(\mathbf{x}_i) - f(\mathbf{x}_j)|}{|\mathbf{x}_i - \mathbf{x}_j|}$
6: **end for**
7: Set $L(\mathbf{x}_i) = \max_j L_{ij}$
8: **end for**

Finally, in order to extend the estimation to a generic location $\mathbf{x}$, it is assumed that $L(\mathbf{x}) = L(\mathbf{x}_{nn})$ where $\mathbf{x}_{nn} = \text{argmin}_{\mathbf{x}_i \in X_n} |\mathbf{x}_i - \mathbf{x}|$. The function $\hat{s}(\mathbf{x})$ mimics the Gaussian Process prediction error and has been designed to quickly increase with increasing distance from an available sample. Moreover, its order of magnitude is comparable to the actual values of the objective function. The adaptive in–fill process is organized as follows: a huge Latin Hypercube Sampling dataset (e.g., 500 times the dimension of the design space) is obtained and the values of the potential of improvement is computed at each point (this requires limited computational effort as the auxiliary function only depends on the surrogate prediction, which is fast to obtain, and on the true objective function values at already collected points); hence, the new sample is located where the maximum value of the auxiliary function is met:

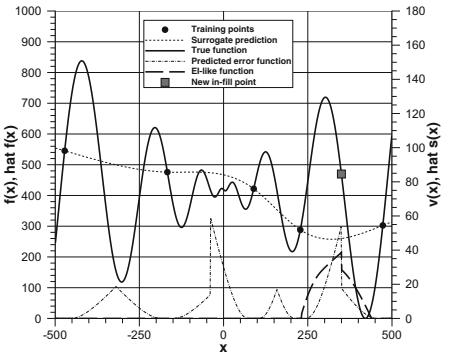
$$\mathbf{x}_{n+1} = \underset{\mathbf{x}}{\text{argmax}}\, v(\mathbf{x}, \hat{f}(\mathbf{x}), X_n, F_{X_n})$$

In order to avoid the duplication of the updating samples when iterating the in–fill process, the seed of the Latin Hypercube is changed at each iteration.

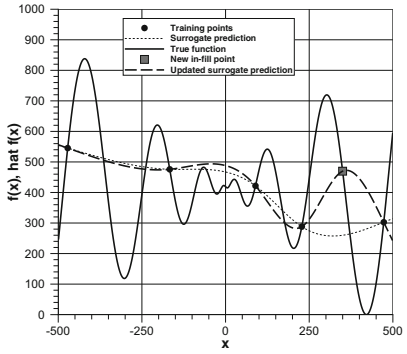
Figure 1 provides an example of surrogate updating by maximization of the EI-like criterion. The one-dimensional Schwefel function is used as test function with 5 initial training points. The trained surrogate (here, a Kriging model) does not capture the local non-linear features of the true function, but a certain trend to predict low values where the true optimum resides is observed (Fig. 1a). The Lipschitz-based prediction error function and the EI-like function are reported in Fig. 1b: by taking the maximum of the EI-like function, a new in-fill point (grey square) is obtained and the surrogate is updated (Fig. 1c). This first iteration seem to not improve the prediction too much: in fact, it provides information about the high non-linearity of the true function around the optimum as the surrogate model now "knows" that the function is rapidly changing in that region. After 10 iterations of the in-fill process,



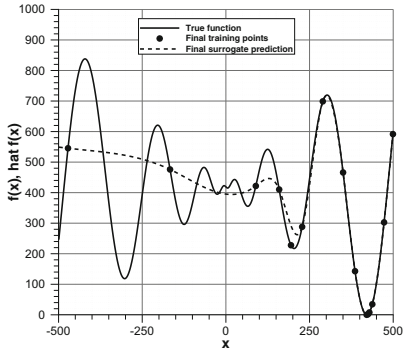
(a) True and surrogate functions $f(\mathbf{x})$, $\hat{f}(\mathbf{x})$ and training dataset $\{X_n, F_{X_n}\}$



(b) Functions $\hat{s}(\mathbf{x})$ and $v(\mathbf{x}, \hat{f}(\mathbf{x}), X_n, F_{X_n})$



(c) New in-fill point and updated surrogate



(d) Updated surrogate with 10 in-fill points

Fig. 1 Example of surrogate updating by maximization of the Lipschitz in-fill criterion on the 1D Schwefel function

the true optimum is perfectly captured as well as the whole trend of the function past $x = 250$ (Fig. 1d).

Surrogate-Based Optimization

The workflow of the surrogate-based shape optimization (SBSO) is depicted in Fig. 2. The method is centered on the surrogate training database which is continuously fed and updated throughout the search and optimization process. As a first step, it is initialized with a space-filling design of experiment (e.g., a Latin Hypercube Sampling or a Latinized Central Voronoi Tessellation): typically, according to literature results and authors past experience, the number of initial samples (n_{apr}) should not exceed

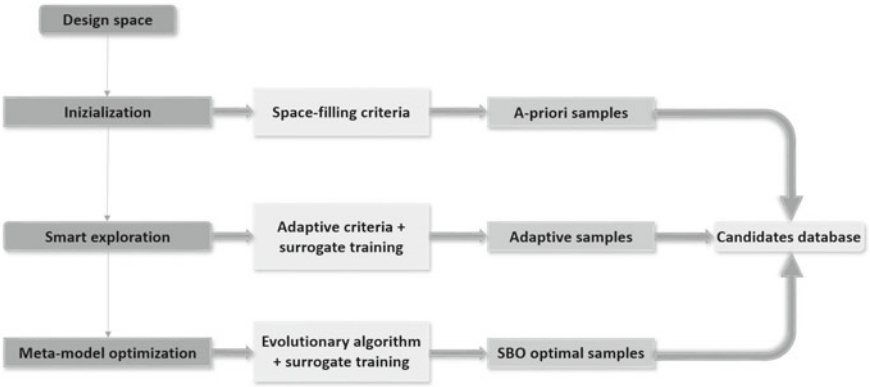


Fig. 2 Workflow of surrogate–assisted optimization

one-third of the total computational budget. The evaluation of the response function corresponding to a given sample is made as follows:

- a geometry parameterization module (CST approach, Kulfan 2008) transforms the design vector (i.e., the training sample) into the actual component shape;
- a batch scripting procedure is launched within ANSYS ICEM CFD package to generate the CAD surface and the volume mesh with fixed sizes and topology;
- a CFD computation is launched with the in-house ZEN CFD flow solver (Catalano and Amato 2003);
- once the simulation has converged, the objective function (usually depending on computed aerodynamic coefficients) and the flow field snapshots are collected according to the specification of the design problem.

As multiple training samples have to be evaluated simultaneously, the process can be executed in parallel to speed up the simulation. Once the evaluation process has finished, the selected surrogate model can be built as described in Section "Surrogate Models".

The workflow in Fig. 2 embeds two internal cycles, namely the adaptive sampling and the optimization update. These iterative phases reflect two different needs: first, providing an improved and reliable model to the optimizer; then, iterating the optimizer to refine the optimum search. The first cycle consists of updating the design solutions database by applying in-fill criteria (as described in Section "Adaptive Sampling Strategy") and providing n_{adpt} new design candidates. The condition to exit from this internal loop is based either on predefined levels of improvement or on computational budget considerations.

The second cycle (database updating by optimization) allows for including n_{opt} sub-optimal samples suggested by sequentially optimizing the meta-model and re-injecting the best candidate in the training database: this phase should lead to the final exploitation of the design space region where the "true" optimum resides. The loop terminates either when the residual of the objective function of the predicted

optima falls below a predefined threshold or when the computational budget limit has been reached. The total computational budget n_{tot} is fixed a-priori and is equal to $n_{tot} = n_{apr} + n_{adpt} + n_{opt}$.

The optimizer consists of an hybrid algorithm implemented within the in-house library ADGLIB (Quagliarella et al. 2004): a genetic algorithm is used for global search and the CMA-ES (Hansen 2006) algorithm acts as a local search operator. During the evaluation of the population, the CMA-ES algorithm is triggered with a predefined activation probability to improve the current best solution.

Numerical Results

The public domain 3rd Drag Prediction Workshop DPW-W1 wing (Epstein et al. 2008) has been selected as the initial geometry for aerodynamic optimization. Reference data for this wing are shown in Table 1. The nominal flow conditions are prescribed at two design points:

1. Mach = 0.76, Reynolds = 5×10^6, $C_{L,0,1} = 0.5$, $C_{D,0,1} = 0.0241$, $C_{M,0,1} = -0.07$
2. Mach = 0.78, Reynolds = 5×10^6, $C_{L,0,2} = 0.5$, $C_{D,0,2} = 0.0279$, $C_{M,0,2} = -0.08$

where $C_{L,0,k}$, $C_{D,0,k}$, $C_{M,0,k}$ are the lift, drag and pitching moment coefficient of the baseline wing at the k-th design point. The objective function to be minimized is:

$$f(\mathbf{x}) = \sum_{k=1}^{2} \frac{1}{2} \frac{C_{D,k} + C_{DM,k} + C_{DL,k}}{C_{L,k}} \frac{C_{L,0,k}}{C_{D,0,k}} \tag{14}$$

$$C_{DM,k} = 0.01 \max(0, C_{M,0,k} - C_{M,k})$$

$$C_{DL,k} = 0.1 \max(0, C_{L,0,k}^2 - C_{L,k}^2)$$

Geometric constraints are also implemented in terms of minimum value of the wing section maximum thickness (=13.5%) and of the beam thickness constraints at two locations along the wing airfoil chord (=12% thickness ratio at 20% wing section chord and 5.9% thickness ratio at 75% wing section chord).

Table 1 Reference data for DPW wing

Wing area	290,322 mm^2
Mean aerodynamic chord	197.55 mm
X_{ref} for moments	154.24 mm (from root l.e.)
Semi-span length	762 mm
Aspect ratio	8.0

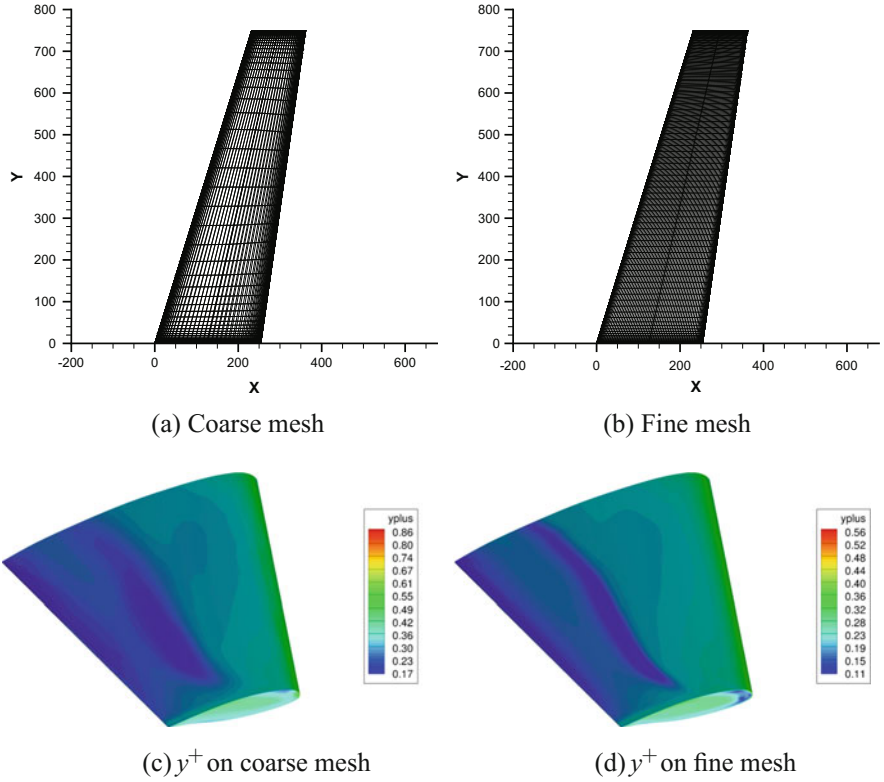
Geometry Parameterization and Mesh Generation

The CST approach allows to identify and isolate the general features which similar shapes have in common (e.g., round/sharp nose, cross section area distribution) and separate the contribution introduced by the real shape change. This "factorisation" is carried out through the definition of a "class" function and a "shape" function, whose product give then the real shape. More details can be found in (Kulfan 2008). In the present case, the wing shape is described by 36 shape variables + 1 variable to modify the twist angle at the wing tip. In order to build the wing shape, three locations along the non-dimensional span length η are selected ($\eta = 0.0, 0.5, 1.0$) and, once given the design weights, the sectional shapes at those three sections are extracted from the analytical CST representation as a set of points; hence, the points are read in ANSYS ICEM CFD and a sequence of parametric commands are executed through a batch script to generate and export the computational mesh. The volume mesh is made of 8 blocks, a family of two grids is defined: the coarse and fine mesh consist respectively of 712,448 cells and 2,959,872 cells. A sketch of the surface mesh distribution is shown in Fig. 3. Both meshes are conceived to respect the $y^+ = \mathrm{O}(1)$ condition, as also shown in the figure where the contour map of y^+ distribution on the wing surface is depicted. The coarse mesh will be used for optimization studies, while the fine mesh will provide more accurate comparisons of the aerodynamic flow for optimized shapes at the end of the optimization process.

Optimization Results

Four different surrogate-based simulations have been carried out and detailed in Table 2. The standard EGO algorithm has been included to set a reference level. The total computational budget is fixed at 500 CFD calls as well as the size of the initial training set is common to all methods and equal to 216. This will allow a fair comparison between the single method capabilities to search the design space with equivalent computational effort. When the present surrogate-based optimization method is used, the EI-like in-fill criterion is adopted for testing purposes.

Figure 4 shows the optimization histories in terms of the progression of the minimum objective function value found in the training database along the iterations. The unit value represents the level of the baseline DPW wing shape. The models have roughly the same pattern, with the POD/RBFN model slightly outperforming the others. The different approach between EGO and the present method is clearly observed: EGO pushes to minimize the objective function from the beginning of the updating phase (i.e., after the initial 216 samples evaluation) having a steady and continuous improvement; on the other hand, the present surrogate-based method achieves a significant contribution to the descent in the final 100 samples, where optimization search is actively working, leaving to the intermediate 184 (adaptive)



(a) Coarse mesh (b) Fine mesh

(c) y^+ on coarse mesh (d) y^+ on fine mesh

Fig. 3 Surface mesh and y^+ distribution on DPW wing surface

Table 2 Optimization setup

Method	Surrogate	In-fill criteria	n_{apr}	n_{adpt}	n_{opt}	Total CFD calls
EGO	Kriging	EI	216	–	284	500
Present SBO	Kriging	EI-like	216	184	100	500
Present SBO	RBFN	EI-like	216	184	100	500
Present SBO	POD/RBFN	EI-like	216	184	100	500

samples the freedom to improve the surrogate quality. At the end of the process, each of the three present SBO methods reaches better results than EGO.

Table 3 propose a comparison of the aerodynamic coefficients and objective function value for all optimal candidates. The keypoint of the optimization task is the drag reduction on DP2, where the improvement is much larger. Slight differences are noticed on pitching moment coefficients as no optimum satisfies the constraint. Indeed, in the minimization problem formulation (Eq. 14), the pitching moment

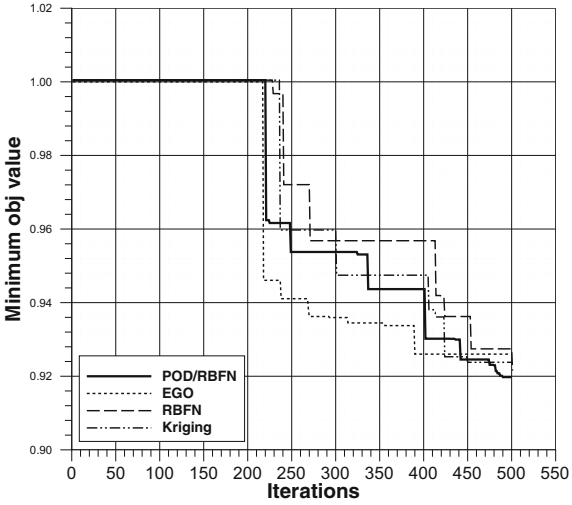


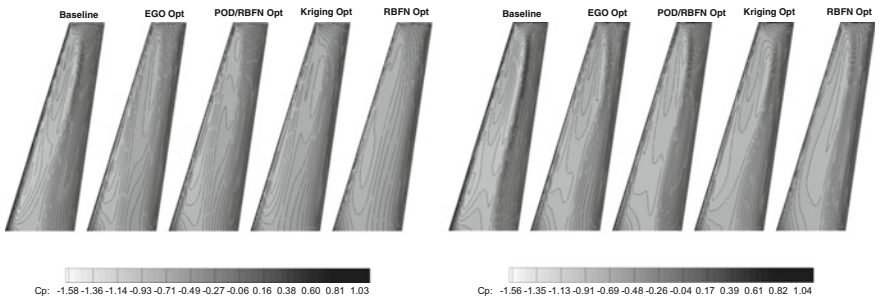
Fig. 4 Convergence histories of surrogate-based optimizations

Table 3 Aerodynamic performances of optimal candidates

Design	DP1			DP2			Obj. value
	$C_{L,1}$	$C_{D,1}$	$C_{M,1}$	$C_{L,2}$	$C_{D,2}$	$C_{M,2}$	
Baseline	0.500	0.0241	–0.0813	0.500	0.0279	–0.0880	1.0
EGO opt.	0.500	0.0231	–0.0942	0.500	0.0244	–0.099	0.926
RBFN opt.	0.500	0.0231	–0.102	0.500	0.0241	–0.108	0.921
Kriging opt.	0.500	0.0232	–0.095	0.500	0.0242	–0.100	0.923
POD/RBFN opt.	0.500	0.0231	–0.086	0.500	0.0243	–0.0918	0.920

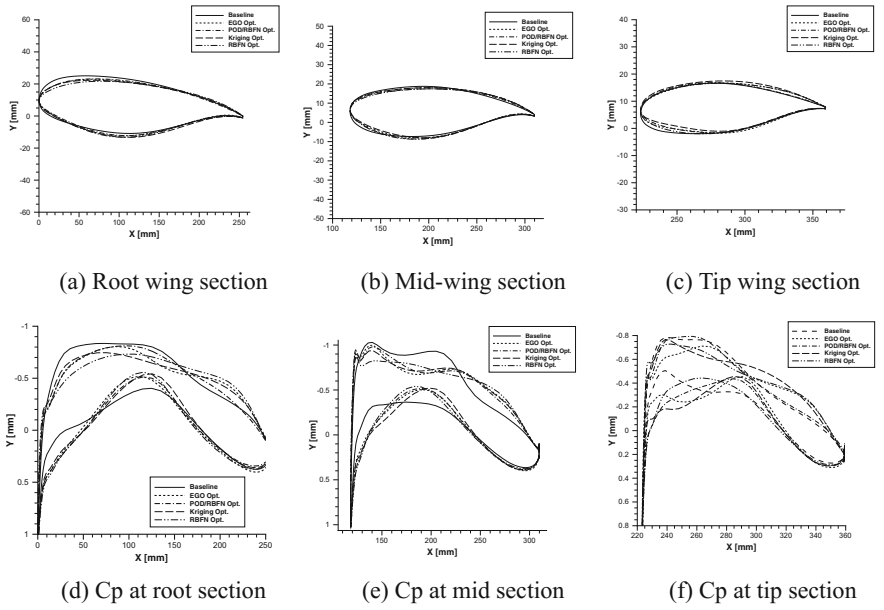
constraint has been implemented as a soft penalty (1 drag counts penalty for 0.01 variation in C_M), hence the method allows to exceed it if the gain in aerodynamic drag is more significant. Anyway, the most interesting result is that all optimal candidates show very similar performances: the relative difference in aerodynamic drag is within 1 count at DP1 and 3 counts at DP3.

Pressure contour maps for selected optimal candidates are depicted in Fig. 5. The inboard wing loading is slightly reduced on design point 1 and a significant decrease of the shock wave intensity is observed on the mid-outboard wing. By comparing the optimal solutions, it is quite evident that EGO and Kriging-based optima are indeed similar as the optimization relied on similar surrogates, even if the adaptive criterion for adding new samples is different. The POD/RBFN model is able to perform slightly better because it is a physics-based approach, i.e. it is fed not only with values of the objective function but mainly with computed flow fields. This peculiar aspect allows to inherit more information related to the nature of the governing equations



(a)DP1 (b)DP2

Fig. 5 Pressure coefficient contour maps



(a) Root wing section (b) Mid-wing section (c) Tip wing section

(d) Cp at root section (e) Cp at mid section (f) Cp at tip section

Fig. 6 Sectional airfoil geometry and Cp distribution

(e.g., flow field structure, shock-wave pattern, boundary layer characteristics) when reconstructing and predicting new solutions. Finally, Fig. 6 proposes a comparison of the local geometry and pressure coefficient solution of the optimal candidates. Three y–constant wing sections are selected, namely at wing root, mid–wing and tip locations. In terms of geometry modifications with respect to the baseline shape, an important reduction of the leading edge curvature is observable and a slight increase of the rear airfoil curvature near the wing tip (probably to recover the lift constraint). The twist angle at the wing tip has also been reduced for wing loading compensation.

Conclusions

The paper proposed a surrogate-assisted methodology suitable to aerodynamic shape optimization. Two scalar–valued surrogates (Kriging, RBFN) and a physics-based meta-model coupling Proper Orthogonal Decomposition and Radial Basis Functions interpolation have been used to predict approximate values of the objective functions throughout the optimization process. The training process has been conceived in three stages, namely a space-filling stage to initialize the surrogate, an adaptive sampling stage in which the model is gradually improved and a final iterative optimization stage where a sequence of improved surrogates are optimized. In the adaptive sampling phase, an in-fill criterion is designed to mimic the Expected Improvement Function maximization by re-formulating the surrogate prediction variance through the estimation of the Lipschitz constant.

An aerodynamic case has been proposed to test the methodology, consisting in the shape optimization of an isolated wing from the AIAA CFD Drag Prediction Workshops with 37 design variables and multi-point conditions. Despite the large scale and the complexity of the case, results are fully satisfactory because of either the obtained improvement (up to 10% on DP2) and the very limited computational cost (only 500 CFD calls).

Such results support the conclusion that surrogate models alone may not provide the right answer within an aerodynamic shape optimization context, especially if transonic viscous flow is considered. However, when coupled to smart adaptive sampling techniques, they allow to catch the basic trends of the objective function without penalizing the design space exploration: indeed, in complex design cases with high non-linearities and multi-modal landscapes, the latter has to be carefully balanced as it may result in unveiling promising regions as well as lead the optimizer to waste time in searching poor solutions.

References

Booker AJ, Dennis JE, Frank PD, Serafini DB, Torczon V, Trosset MW (1999) A rigorous framework for optimization of expensive functions by surrogates. Struct Multi Optim 17:1–13. https://doi.org/10.1007/BF01197708

Braconnier T, Ferrier M, Jouhaud J-C, Montagnac M, Sagaut P (2011) Towards an adaptive POD/SVD surrogate model for aeronautic design. Comput Fluids 40(1):195–209

Catalano P, Amato M (2003) An evaluation of RANS turbulence modelling for aerodynamic applications. Aerosp Sci Technol 7:493–509

da Silva Santos CH, Goncalves MS, Hernandez-Figueroa HE (2010) Designing novel photonic devices by bio-inspired computing. IEEE Photonics Technol Lett 22(15):1177–1179

Epstein B, Jameson A, Peigin S, Roman D, Vassberg J, Harrison N (January, 2008) Comparative study of 3D wing drag minimization by different optimization techniques. In: 46th AIAA Aerospace Sciences Meeting and Exhibit. American Institute of Aeronautics and Astronautics

Fasshauer GE, Zhang JG (2007) On choosing "optimal" shape parameters for RBF approximation. Numer Algorithms 45(1):345–368

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

Gutmann HM (2001) A radial basis function method for global optimization. J Global Optim 19:201–227

Hansen N (2006) The CMA evolution strategy: a comparing review. In: Lozano J, Larranaga P, Inza I, Bengoetxea E (eds) Towards a new evolutionary computation. Advances on estimation of distribution algorithms, Springer, Berlin, pp 75–102

Iuliano E (2011) Towards a POD-based surrogate model for CFD optimization. In: Proceedings of the ECCOMAS CFD & optimization conference. Antalya, Turkey

Iuliano E, Quagliarella D (2011) Surrogate-based aerodynamic optimization via a zonal pod model. In: Proceedings of the EUROGEN 2011 Conference. Capua, Italy

Iuliano E, Quagliarella D (2013) Proper orthogonal decomposition, surrogate modelling and evolutionary optimization in aerodynamic design. Comput Fluids 84:327–350

Jones DR, Schonlau M, Welch WJ (1998) Efficient global optimization of expensive black-box functions. J Global Optim 13:455–492

Kulfan BM (2008) Universal parametric geometry representation method. J Aircr 45(1):142–158

Mack Y, Goel T, Shyy W, Haftka R (2007) Surrogate model-based optimization framework: a case study in aerospace design. Springer, Berlin, pp 323–342

Martin JD, Simpson TW (2005) Use of kriging models to approximate deterministic computer models. AIAA J 43(4):853–863

Quagliarella D, Iannelli P, Vitagliano PL, Chinnici G (2004) Aerodynamic shape design using hybrid evolutionary computation and fitness approximation. In: AIAA 1st intelligent systems technical conference. American Institute of Aeronautics and Astronautics (AIAA), Chicago, IL (AIAA Paper 2004-6514)

Richardson JA, Kuester JL (1973) Algorithm 454: the complex method for constrained optimization [e4]. Commun. ACM 16(8):487–489

Rippa S (1999) An algorithm for selecting a good value for the parameter c in radial basis function interpolation. Adv Comput Math 11(2):193–210

Robinson T, Willcox K, Eldred M, Haimes R (September, 2006) Multifidelity optimization for variable-complexity design. In: 11th AIAA/ISSMO multidisciplinary analysis and optimization conference. American Institute of Aeronautics and Astronautics

Tenne Y, Armfield SW (2008) A versatile surrogate-assisted memetic algorithm for optimization of computationally expensive functions and its engineering applications. Springer, Berlin, pp 43–72

Viana FAC, Haftka RT (2012) Watson LT Efficient global optimization algorithm assisted by multiple surrogate techniques. J Global Optim 56(2):669–689