AIAA JOURNAL
Vol. 48, No. 5, May 2010

Comparison of Surrogate Models in a Multidisciplinary Optimization Framework for Wing Design

Ricardo M. Paiva,* André R. D. Carvalho,* Curran Crawford,† and Afzal Suleman‡
University of Victoria, Victoria, British Columbia, Canada

DOI: 10.2514/1.45790

The replacement of the analysis portion of an optimization problem by its equivalent metamodel usually results in a lower computational cost. In this paper, a conventional nonapproximative approach is compared against three different metamodels: quadratic-interpolation-based response surfaces, Kriging, and artificial neural networks. The results obtained from the solution of four different case studies based on aircraft design problems reinforces the idea that quadratic interpolation is only well-suited to very simple problems. At higher dimensionality, the usage of the more complex Kriging and artificial neural networks models may result in considerable performance benefits.

Nomenclature

ANN	=	artificial neural networks
$b/2$	=	Wing semispan, m
$\mathbf{c}$, c_i	=	Coefficients for polynomial interpolation
c_{bs}	=	Wing breakstation chord, m
c_{root}	=	Wing root chord, m
c_{tip}	=	Wing tip chord, m
$f(\mathbf{x})$	=	Regression model (Kriging)
$g(\mathbf{x})$	=	Constraint function
L	=	Wing lifting force, N
m_{wing}	=	Wing mass, kg
n_{DV}	=	Number of design variables
n_s	=	Number of samples
n_t	=	Number of terms in polynomial interpolation/ regression approximation
$q_k(\mathbf{x})$	=	Values of regression functions at sample locations (Kriging)
$\mathcal{R}(\mathbf{w}, \mathbf{x}, \theta)$	=	Correlation model (Kriging)
S_{ref}	=	Wing reference surface area, m^2
$\mathbf{s}_k$	=	Vector of independent-variable samples (Kriging)
$\mathbf{X}$	=	Vector of design variables
$\mathbf{x}$, x_i	=	Vector of design variables
x_j^m	=	Input vector for m th layer (ANN)
W_j	=	ANN input weights
W_{ij}^m	=	Weight matrix for the m th layer (ANN)
W_{other}	=	Aircraft weight (minus wings), N
$\hat{y}$	=	Function approximation
y_{bs}	=	Breakstation spanwise location, m
$\mathbf{y_s}$	=	Vector of function samples
$z(\mathbf{x})$	=	Interpolated residuals (Kriging)
α	=	Angle of attack, deg
β	=	ANN input bias
β_i^m	=	Bias vector for m th layer
β_k	=	Least-squares weights
γ	=	Activation function (ANN)
γ'_{ij}	=	Jacobian matrix of the activation functions for a layer (ANN)
δ_i^m	=	Sensitivity of the m th layer (ANN)
η	=	Learning rate (ANN)
θ, θ_j	=	Correlation fitting parameters (Kriging)
θ_{bs}	=	Twist at wing breakstation, deg
θ_{tip}	=	Twist at wing tip, deg
λ_k	=	Kriging weights
Λ_{LE}	=	Wing leading-edge sweep, deg
ρ	=	Kreisselmeier–Steinhauser constraint aggregation parameter
$f(\sigma_i, \sigma_{\text{yield}})$	=	Failure criteria

Presented as Paper 2203 at the 50th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference 17th AIAA/ASME/AHS Adaptive Structures Conference 11th AIAA Non-Deterministic Approaches Conference 10th AIAA Gossamer Spacecraft Forum 5th AIAA Multidisciplinary Design Optimization Specialists Conference, Palm Springs Convention Center/Wyndham Palm Springs Palm Springs, California, 4–7 May 2009; received 2 June 2009; revision received 18 November 2009; accepted for publication 15 December 2009.

*MASc, Ph.D. Candidate, Department of Mechanical Engineering. Student Member AIAA.

†Assistant Professor, Department of Mechanical Engineering. Member AIAA.

‡Professor, Department of Mechanical Engineering; Principal Investigator, IDMEC IST, Lisbon, Portugal. Associate Fellow AIAA.

I. Introduction

DESPITE the advancements made in computer technology, it is a fact that some computational problems still remain quite cumbersome to solve, and these difficulties are further accentuated when trying to solve them in a single or multidisciplinary design optimization environment. By replacing the computationally expensive analyzers with suitable surrogate models/metamodels, the optimization process can be greatly accelerated, even if at the expense of a loss in fidelity [1]. In this paper, an evaluation of some well-known methods to create such surrogate models is performed, with applications based on simple aircraft-design problems. The three approximation models implemented in this work are quadratic interpolation, Kriging, and artificial neural networks (ANN), which are covered in more detail in Sec. II.

Several applications of these approaches to optimization problems are present in the literature. For instance, Giunta applied both polynomial models and Kriging to the optimization of a high-speed civil transport design, while employing a multiple-fidelity approach [2]. Kriging metamodels, coupled with genetic algorithms, have also seen use when solving multiobjective and multidisciplinary optimization problems, as in the work of Kanazaki et al. [3] and Kumano et al. [4]. In a rare comparison of such different types of metamodels, Matias et al. used Kriging and ANN and suggested that Kriging has the advantage of not requiring as much tweaking as neural networks [5]. Willmes et al. go even further, comparing these approximations with standard use of evolutionary algorithms, effectively finding no advantages in their application to simple test problems [6].

In light of this evidence, the main objective of the current work is to assess the performance benefits (or lack thereof) in using these approximations in conjunction with an multidisciplinary design

optimization (MDO) tool tailored for the preliminary design of aircraft wings [7]. The modular nature of this tool, which is described in further detail in Sec. III, facilitates the integration of the modules required for building the surrogate models. The resulting integrated approach is coupled with an adaptive sampling strategy, which is described in Sec. II.D.

II. Surrogate models

Three response-surface/metamodel generating methods are used in the current work: quadratic interpolation, Kriging, and neural networks. The following sections serve as an introduction to each of them in some detail.

A. Quadratic interpolation

Quadratic-interpolation-based surrogate models are among the fastest available (in terms of execution speed at both creation and evaluation time), and their implementation is rather trivial [2]. As the name implies they involve the fitting of a second-order polynomial to the function of interest (in some applications, incomplete versions of these polynomials are used, as some cross-coupling terms are excluded).

The interpolating polynomial is therefore written as

$$\hat{y} = c_0 + \sum_{j=1}^{n_{\mathrm{DV}}} c_j x_j + \sum_{k=1}^{n_{\mathrm{DV}}} \sum_{j=k}^{n_{\mathrm{DV}}} x_k x_j c_{1+n_{\mathrm{DV}}+\sum_{m=1}^{k-1}(n_{\mathrm{DV}}-m+1)+j-k} \quad (1)$$

The total number of polynomial coefficients n_t is in this case equal to the number of samples to be taken n_s:

$$n_t = \frac{(n_{\mathrm{DV}}+1)(n_{\mathrm{DV}}+2)}{2} = n_s \quad (2)$$

The analytical derivative of the quadratic polynomial is readily calculated as

$$\frac{\mathrm{d}\hat{y}}{\mathrm{d}x_i} = c_i + \sum_{\substack{k=1 \\ k\neq i}}^{n_{\mathrm{DV}}} c_{F(i,k)} x_k + 2x_i c_{1+n_{\mathrm{DV}}+\sum_{m=1}^{i-1}(n_{\mathrm{DV}}-m+1)} \quad (3)$$

$$F(i,k) = \begin{cases} 1 + n_{\mathrm{DV}} + (\sum_{m=1}^{k-1} n_{\mathrm{DV}} - m + 1) + i - k, & k < i \\ 1 + n_{\mathrm{DV}} + (\sum_{m=1}^{i-1} n_{\mathrm{DV}} - m + 1) + k - i, & k > i \end{cases} \quad (4)$$

The fact that sensitivities may be computed analytically is an advantage of such models. Furthermore, the natural smoothness of the polynomial filters out local disturbances in the interpolated function (which may be due to numerical errors, mesh adaptation, etc.). This ensures that, in most cases, these sensitivities do represent the global design trends (though the method may fill in local extreme). Unfortunately, this also means that the ability to approximate true local variations in the function to be modeled is impaired, if in fact these exist.

The polynomial coefficients $\mathbf{c}$ are obtained by solving the linear system of equations, which equates the values of the function at the sample points $\mathbf{y_s}$ to the value of the polynomial at those same points, thereby employing exact interpolation:

$$\mathbf{y_s} = \mathbf{X}\mathbf{c} \quad (5)$$

$$\mathbf{X} = \begin{bmatrix} 1 & x_1^{(1)} & x_2^{(1)} & \cdots & (x_1^{(1)})^2 & x_1^{(1)}x_2^{(1)} & \cdots & (x_{n_{\mathrm{DV}}}^{(1)})^2 \\ \vdots & & \ddots & & \vdots & & & \vdots \\ 1 & x_1^{(n_s)} & x_2^{(n_s)} & \cdots & (x_1^{(n_s)})^2 & x_1^{(n_s)}x_2^{(n_s)} & \cdots & (x_{n_{\mathrm{DV}}}^{(n_s)})^2 \end{bmatrix} \quad (6)$$

$$\mathbf{c} = [\, c_0 \quad c_1 \quad \cdots \quad c_{n_s-1} \,]^T \quad (7)$$

$$\mathbf{c} = \mathbf{X}^{-1}\mathbf{y_s} \quad (8)$$

B. Kriging

Kriging is a statistical interpolation method, initially designed to predict the size of oil reserves in the mining industry. Originally introduced by Krige (hence, the designation) in 1951, this field of geostatistics would not be established until a decade later [8].

Some recent examples of the use of Kriging to develop surrogates to be used in design optimization are the work of Huang [9], Lee and Park [10], and Simpson et al. [11]. Several types of Kriging have been employed thus far. Some of the most well-known variants are simple Kriging, ordinary Kriging, and regression Kriging. The latter is the one which was chosen for use in the present work. Regression Kriging is a hybrid approach between a (usually polynomial) regression and ordinary Kriging models. It is based on the assumption that the value of a given multivariable function $y(\mathbf{x})$ can be decomposed into deterministic and stochastic components:

$$y(\mathbf{x}) = f(\mathbf{x}) + z(\mathbf{x}) \quad (9)$$

$f(\mathbf{x})$ represents the deterministic model, $z(\mathbf{x})$ are the interpolated residuals, and $\mathbf{x}$ is a location of interest, where no samples were taken. While the regression model describes the behavior of the function on a global level, the local discrepancies between such a model and the actual function are taken into account through the Kriging model. The global regression model takes the form

$$f(\mathbf{x}) = \sum_{k=1}^{n_t} \beta_k q_k(\mathbf{x}) \quad (10)$$

q_k are a set of predictor functions, n_t is the total number of terms used in the regression, and β_k are their respective weights determined by means of fitting the sample data; in the present case, generalized least squares is employed. Least-squares fitting of the polynomial approximations in Sec. II.A was not pursued because of diminishing returns of a nonlocally refined and noninterpolating approximation. The current implementation supports polynomials up to second order as the prediction functions.

On the other hand, $z(\mathbf{x})$ is assumed to have zero mean and covariance defined by

$$E[z(\mathbf{x})z(\mathbf{w})] = \sigma^2 \mathcal{R}(\mathbf{w}, \mathbf{x}, \theta) \quad (11)$$

where σ is the standard deviation and $\mathcal{R}(\mathbf{w}, \mathbf{x}, \theta)$ is a correlation function defined between points $\mathbf{x}$ and $\mathbf{w}$ (see Table 1), fitted to sample data by means of the parameters θ. In the Kriging approximation, the samples are weighted in the following manner:

$$\hat{y}(\mathbf{x}) = \sum_{k=1}^{n_s} \lambda_k y(\mathbf{s}_k) \quad (12)$$

where $y(\mathbf{s}_k)$ are the values of the function at the sample points $(\mathbf{s}_k)$ and λ_k are the Kriging weights that need to be estimated. For the purposes of implementing a Kriging approximation of the objective and constraint functions in the MDO problems, a MATLAB toolbox (DACE) was converted to C#/.NET and hence designated DACE.NET. The detailed procedure of obtaining the weights in regression Kriging escapes, however, the scope of this paper, as it is thoroughly explained in the manual accompanying this toolbox [12] as well as in other publications [8,13].

Table 1 Examples of correlation models for Regression Kriging

Name	Correlation model
Exponential	$e^{-\lvert w_j - x_j\rvert\theta_j}$
General exponential	$e^{-\lvert w_j - x_j\rvert^p\theta_j}, 0 < p \leq 2$
Gaussian	$e^{-(w_j - x_j)^2\theta_j}$
Spherical	$1 - 1.5\xi_j + 0.5\xi_j^3, \xi_j = \min\{1, \theta_j\lvert w_j - x_j\rvert\}$
Linear	$\max\{0, 1 - \theta_j\lvert w_j - x_j\rvert\}$
Cubic	$1 - 3.0\xi_j + 2.0\xi_j^3, \xi_j = \min\{1, \theta_j\lvert w_j - x_j\rvert\}$

Derivatives from the Kriging model are also easy to obtain since the derivatives from the regression and correlation functions are well-known [8,12].

C. ANN

ANN were first theorized by McCulloch and Pitts in 1943 [14]. Based on their knowledge of the operation of organic brains, McCulloch and Pitts established several network configurations for logical neurons. In the years that followed, ANN were studied in great detail by the mathematical and computational analysis community. In turn, this lead to major breakthroughs such as the development of the first learning rule by Hebb in 1949, the first perceptron by Rosenblatt in 1958, and the adaptive linear element by Widrow and Hoff in 1960 [15]. Nevertheless, in the following decades, there was a drastic decrease in interest in ANN since additional developments in the area would have required computational power not yet available at the time. This situation lasted until the early 1980s, when digital microprocessors began to see widespread use. In light of these technological achievements, ANN research regained some of the momentum it once had with the development of the associative memory network, by Hopfield, and with the self-organizing map, by Kohonen, both developed in 1982 [15,16] Nowadays, there are over 20 different types of ANN used in a vast range of applications ranging from nonlinear control to data mining.

The basic unit of ANN is the artificial neuron, which in its simpler form is referred to as a *perceptron*. The perceptron (see Fig. 1) is a function with two components: a weighted summation of the inputs [Eq. (13)] and an activation function [Eq. (14)] [15,17]:

$$n = W_j x_j + \beta \tag{13}$$

$$y = \gamma(n) \tag{14}$$

where x_j are the inputs of the perceptron, β is a bias, W_j are the weights of each input, and γ and y are the activation function and the result of the weighted summation, respectively.

The activation function can be any monotonic function, but the following are the most commonly used: origin-crossing linear functions, hyperbolic tangents, logistic functions, or the sign function. Usually, linear functions are used in the input and output layers of a network, the sign function is used for biological-inspired applications or in digital networks, the hyperbolic tangent is preferentially used in system identification and control, and the logistic function is used in data mining [15,17]. There are other, more complex, types of artificial neurons (e.g., radial basis perceptrons), but in this work only the regular perceptrons were employed.

The multilayer perceptron (MLP) is one of the configurations proposed by McCulloch and Pitts and is the most versatile and simple network structure. As the name implies, the MLP is composed of a series of sequentially connected layers of perceptrons. A layer is defined as a group of perceptrons sharing a common set of inputs,

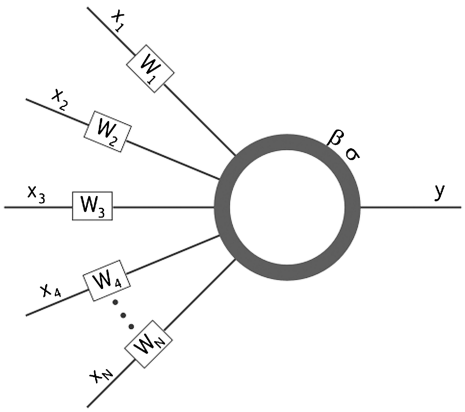

Fig. 1 Perceptron: simplest form of the artificial neuron.

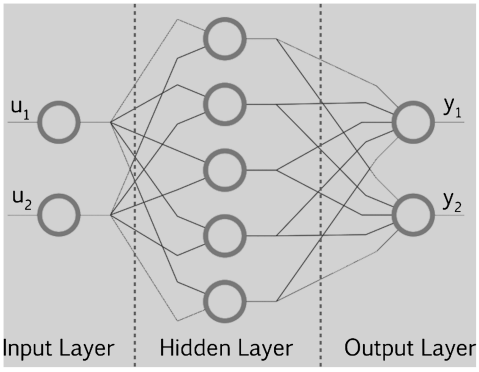

Fig. 2 Multilayer perceptron showing the three types of layers.

and, in the case of the MLP, there cannot be intralayer connections between perceptrons (Fig. 2) [15].

An MLP is divided into three functional regions: the input layer, the hidden layers, and the output layer [15].

The hidden layers are where the bulk processing of the network takes place. An MLP can have several hidden layers, however the required number of layers or the number of perceptrons per layer is dependent on the application. These parameters are very important, because they will define the overall performance of the network and must be selected according to the complexity of the problem. There are no established methods to determine the parameters for a certain application of an MLP, and usually they must be set by a trial-and-error process or by discrete-optimization-capable methods, (e.g., genetic algorithms).

Usually, for an MLP with a single hidden layer, if the number of perceptrons is too small, the network will not converge or will converge to a suboptimal configuration. As one increases the number of perceptrons, the network will systematically converge to better configurations, until a point where the network becomes too complex and starts to overfit the training data, losing quality. Because of this behavior, the Kolmogorov theorem states that there is always an optimal set of parameters for which a neural network perfectly interpolates any given application [15].

The main advantage of neural networks is their adaptive nature. The process by which an MLP learns is called the back-propagation algorithm. The back-propagation algorithm systematically applies a gradient-based optimization algorithm to each layer of the MLP. The back propagation is divided in two parts: in the first the sensitivities for each layer are evaluated, starting from the output layer and ending on the first hidden layer; in the second part the weights are updated according to the computed sensitivities, a process that starts in the first hidden layer and ends on the output layer.

The mathematical equations of the algorithm are highly dependent on the optimization algorithm being used. For the steepest descent, which is the most simple and most common algorithm, Eqs. (15) and (16) represent the first and second parts, respectively:

$$\delta_i^m = \gamma'_{ij} W_{jk}^{m+1} \delta_k^{m+1} \tag{15}$$

$$\begin{cases} W_{ij}^m = W_{ij}^m + 2\eta \delta_i^m x_j^m \\ \beta_i^m = \beta_i^m + 2\eta \delta_i^m \end{cases} \tag{16}$$

where δ_i^m is the sensitivity for neuron i of layer m; γ'_{ij} is the Jacobian matrix of the activation functions of the layer; W_{ij}^m and β_i^m are, respectively, the weight matrix and bias vector for the layer; and η is the learning rate.

The implementation of the neural network model was done in C++. The neural network is enclosed in a class that has methods for training and testing the network. From these methods, one can obtain the results of the surrogate model and the corresponding derivative for a given input point. To be used with the MDO tool, the class was implemented in a dynamic library, thus maximizing portability.

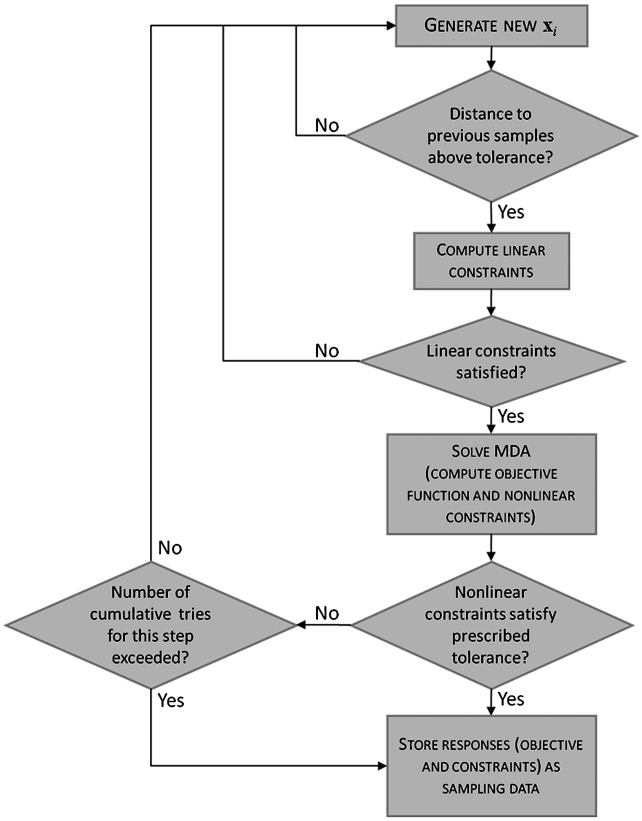

Fig. 3 Sampling algorithm with constraint-based filtering.

D. Sampling

For the purposes of generating the samples required to build the surrogate models, both random and Latin hypercube samplers were created. The process of creating the sample space, though customizable by the user, is oriented toward maximizing the number of samples lying within the constrained design space while minimizing the number of actual objective/constraints functions evaluations. To that end, a newly generated sample (which is already within the "hard" bounds) is submitted to the tests illustrated in Fig. 3, in which the distance to previously taken samples and linear constraints are computed first to avoid a possibly superfluous function evaluation. In the case of nonlinear constraints, after a number of failed tries, the algorithm finally includes the constraint violating point into the sampling set. The maximum number of failed tries is a user-defined setting that, depending on the problem, may have a considerable impact on performance since it prevents the optimization process from being stalled at the sampling phase (a value of 20 is used by default). This situation frequently occurs near the intersection of two or more constraints as illustrated in Fig. 4 (the size of the clearance areas is exaggerated, but the number of samples is reduced in order to favor understandability). To further ameliorate this issue, there is the option in the algorithm to include a tolerance for infeasible points (with respect to nonlinear constraints, i.e., $g(\mathbf{x}) \leq \epsilon$, $\epsilon > 0$, instead of $g(\mathbf{x}) \leq 0$).

Because an initial model will generally provide poor coverage and for that reason the optimum found may still be far from a true optimum, an adaptive trust region can be defined around this point so that the sample is updated and further refined around this area of interest. Hence, the sampling procedure described above is executed only once on the overall design space, and subsequently it is confined to the limits of the trust region, which is also illustrated in Fig. 4.

III. MDO tool

This section will serve as an overview of the MDO tool developed, focusing on the description of the modules already implemented as well as on the user interface itself.

Because user interaction is of prime concern in design optimization, particularly in the initial stages, a graphical interface (GUI) for the tool was developed. Its advantages in terms of the setup of optimization problems and postprocessing of results have become self-evident during development and initial testing. The GUI was developed using the C# object-oriented language, as it facilitates the process of setting up the various elements of a graphical interface (windows, menus, buttons, etc.) as compared with Visual C++, for instance.

One of the requirements for such an interface was to make it modular, so that the various analyzer/optimizer modules could be easily swapped for higher fidelity and/or more capable versions. To that end, provisions were made so that these modules can be accompanied by a suitable interface that serves as a bridge between the user interface itself and the underlying analyzer module. Furthermore, if the module in question is compiled as an application extension (dynamic-link library), input and output procedures are performed much faster since all data is exchanged through system memory (RAM), rather than through files (and therefore using the much slower hard drives). More important, the interface incorporates a self-updating, interactive 3D viewer with a detailed representation of the wing, so that the user is fully aware at all times during the optimization procedure.

The interface is divided into several tabs representing each of the modules, the first of which is the geometry module (Fig. 5), in which the contour of the wing is defined. At this point, three custom airfoils may be defined from which the wing is extruded according to user-defined planform variables (semispan, chords at root, break station, and tip, among others).

The next tab comprises the aerodynamics module (Fig. 6), in which the main options allow the definition of flight conditions and mesh parameters for the underlying vortex lattice method (VLM) code [18]. Upon computation, the results are shown in terms of a pressure-distribution contour plot and major aerodynamic coefficients (C_L, C_D, C_M, and their derivatives, should the user require them).

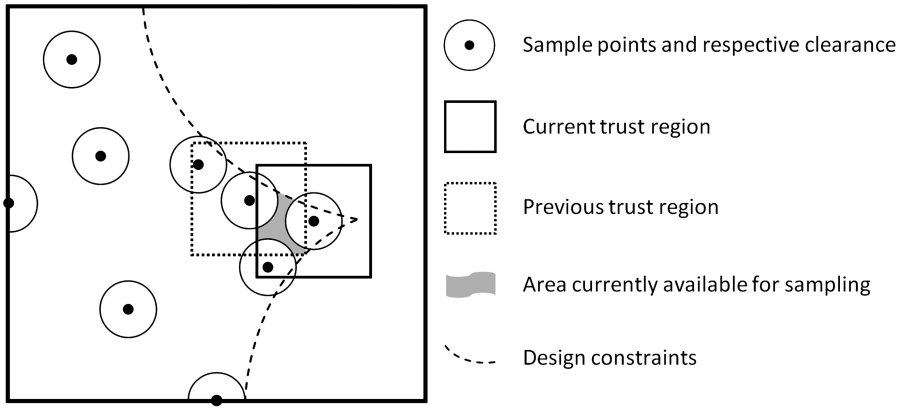

Fig. 4 Sampling region in the vicinity of constraints.

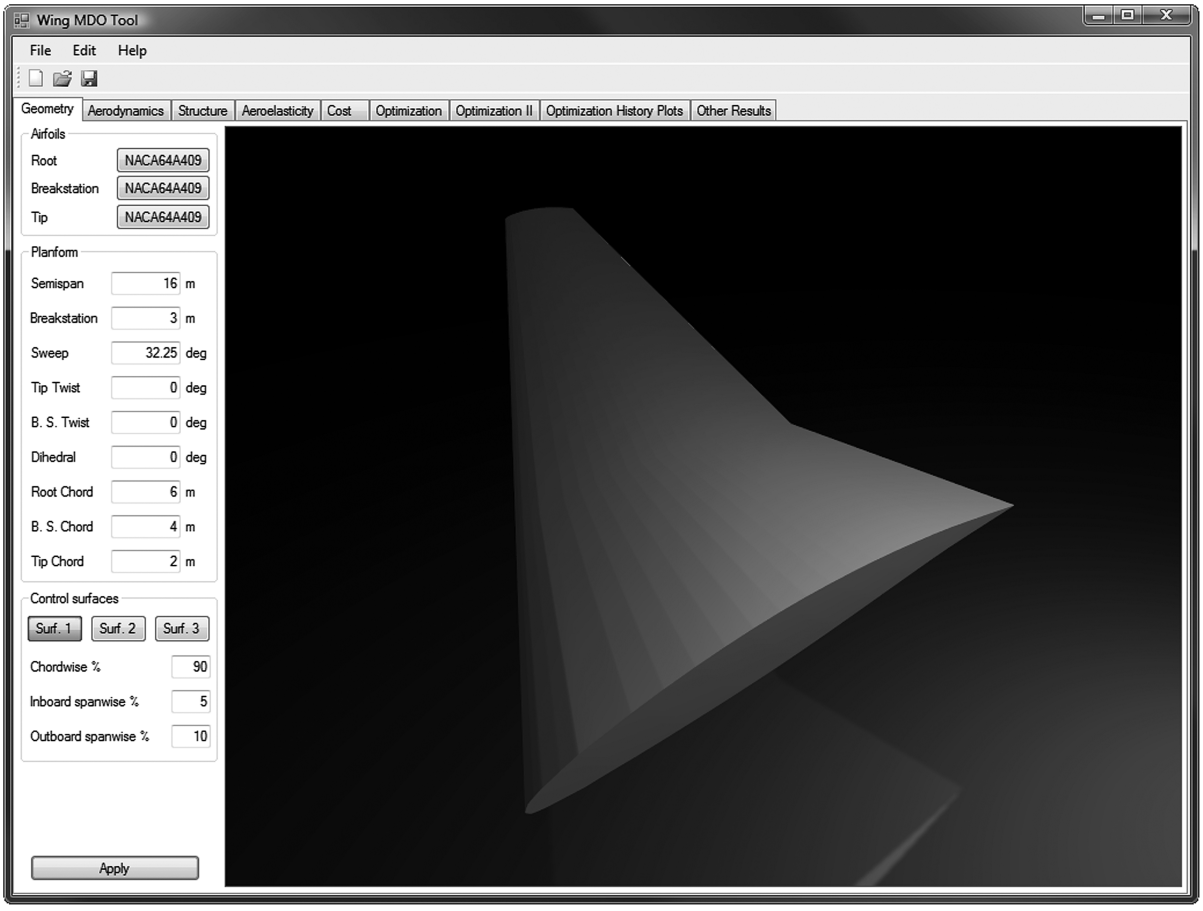

Fig. 5 Geometry generation tab.

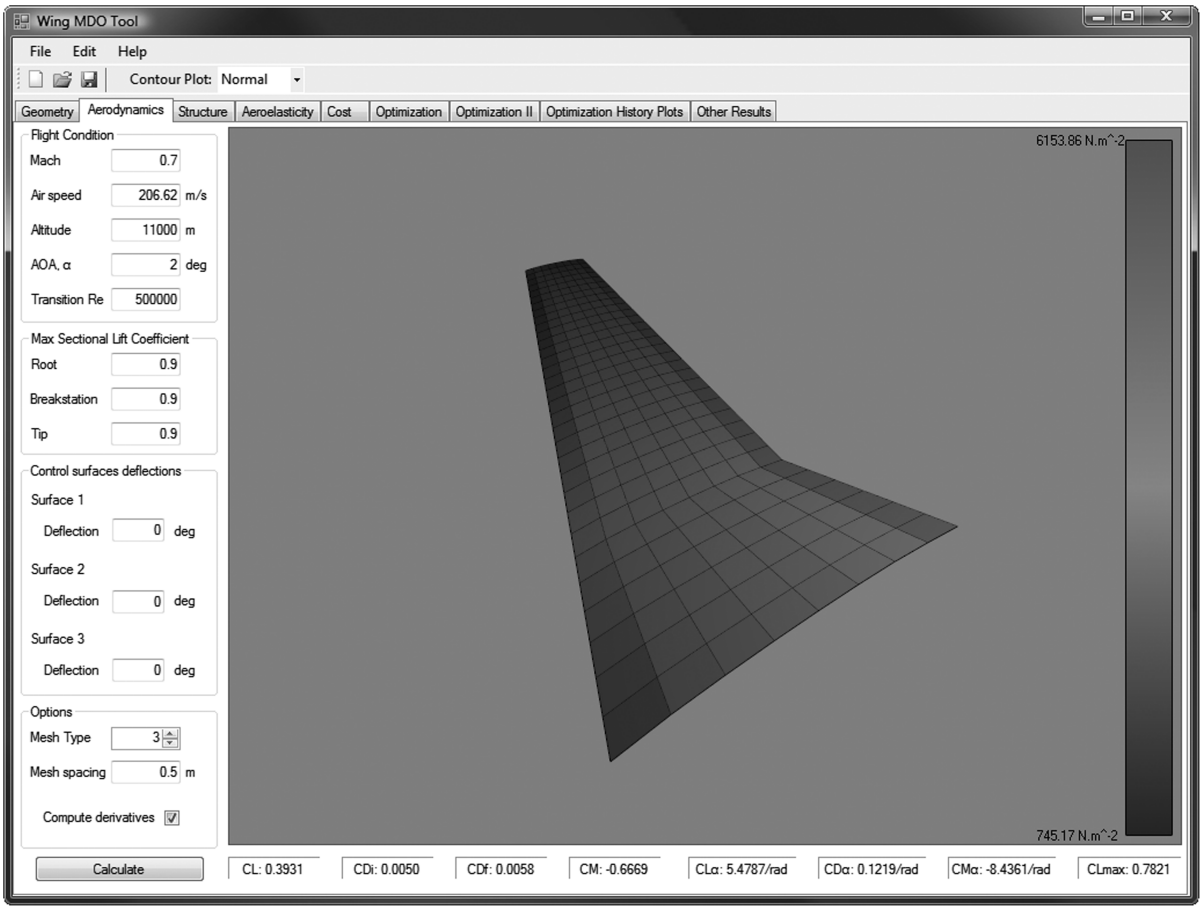

Fig. 6 Aerodynamics tab.

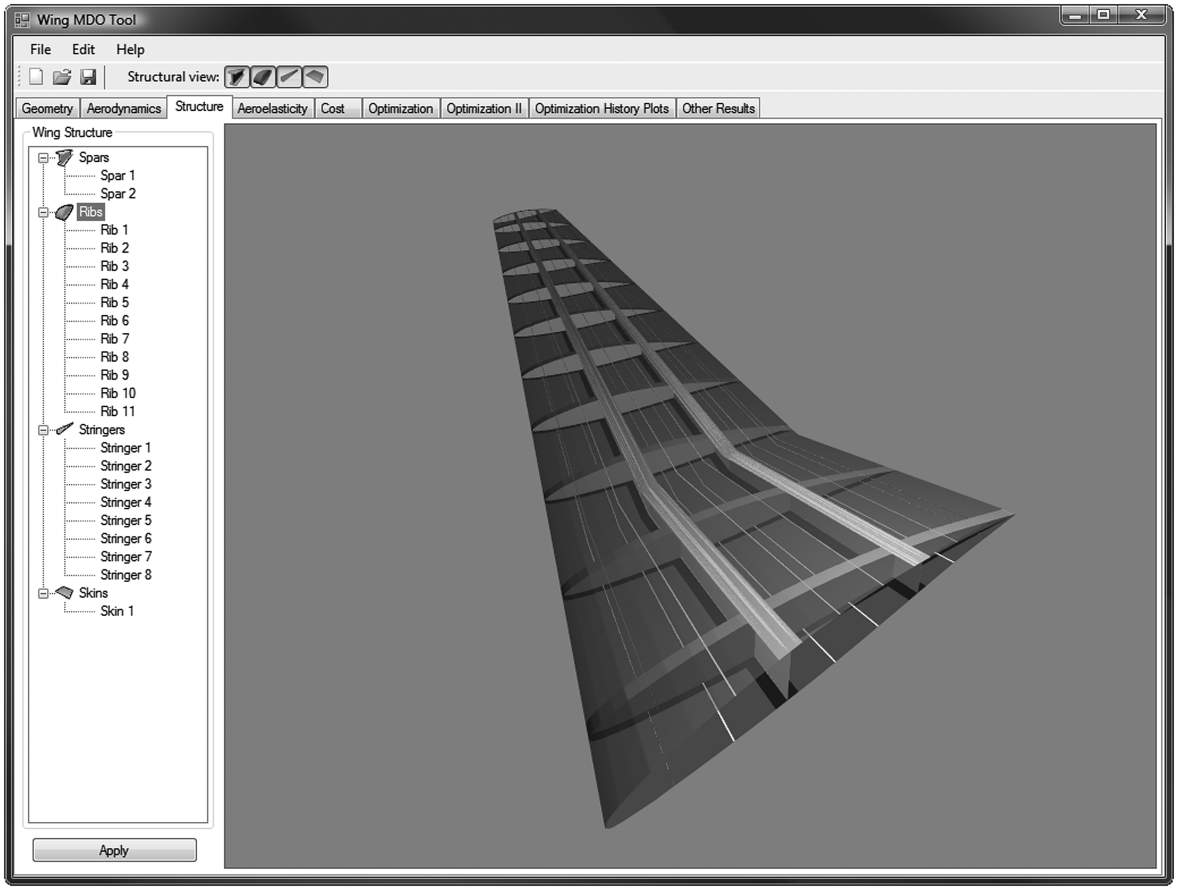

Fig. 7 Wing-structure definition tab.

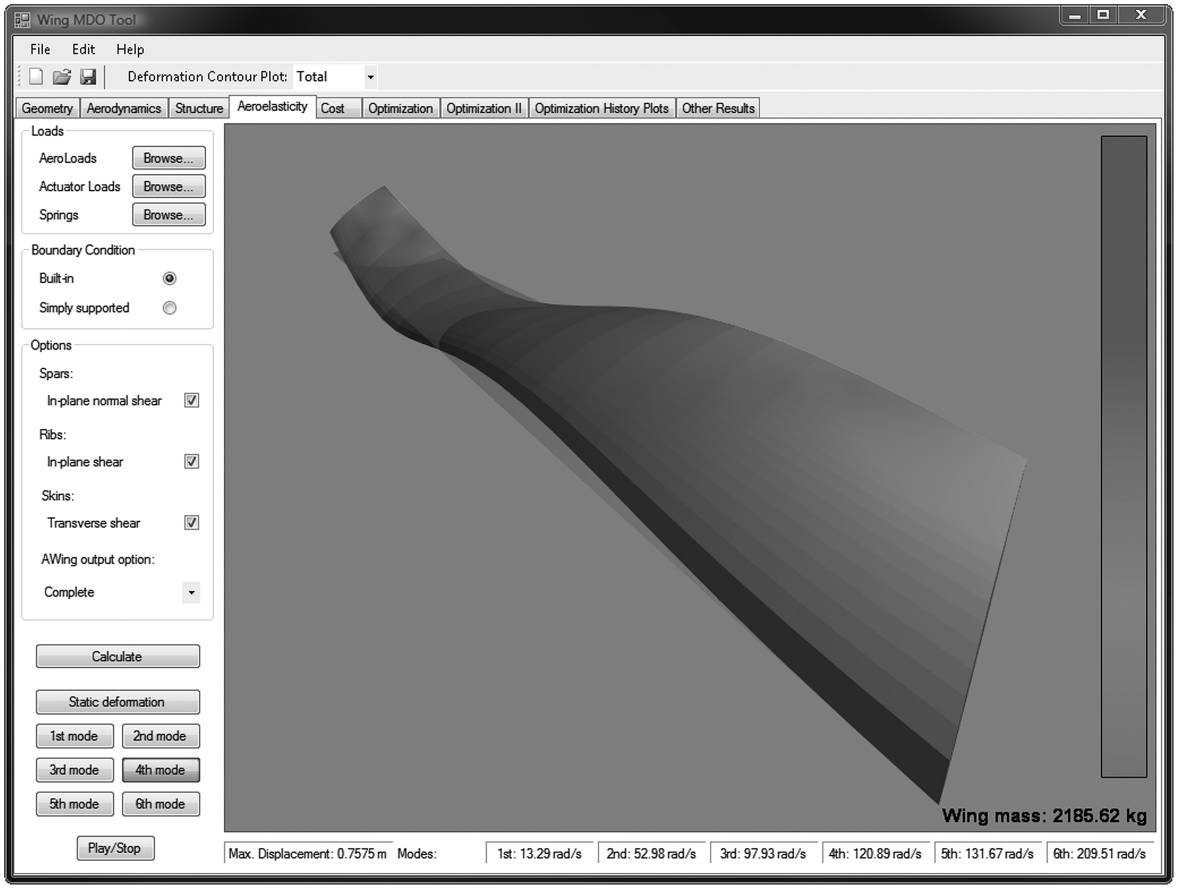

Fig. 8 Aeroelasticity tab.

The VLM code Zephyr was developed specifically for the current implementation of the tool, and has support for compressible flow (Prandtl–Glauert transformation). It also includes a skin-friction estimate, which uses a combination of the Eckert reference temperature method for laminar flow and the van Driest 2 formula for turbulent flow [19]. This module has been validated through comparison against other VLM codes and a Reynolds-averaged Navier–Stokes) code: OVERFLOW [7,20]. Support has also been added for the definition of control surfaces.

Figure 7 depicts the definition of the wing's structure. This is accomplished through a treeview control comprising four main categories: spars, ribs, stringers, and skin panels. Popup menus allow the definition of individual-components positioning, section parameters, and material properties. The structural module is based on an equivalent plate model theory as proposed by Kapania and Liu [21], Giles [22], Livne [23], and Livne and Navarro [24]. As implemented (in a legacy code, designated Awing), this method allows the definition of a detailed wing structure composed of spars, ribs, stringers, and skin panels.

The code is, for the moment, limited to the study of trapezoidal wings, which has motivated its impending replacement for a newer, finite-element-based module. The fluid structure interaction is also simply one-way, wherein the aerodynamics loads (point loads at the vortex-lattice elements control point) are passed on to the structural analyzer for computation of the static deformation. Semi-empirical relations are then used to estimate other performance figures of interest, such as flutter speed, for instance.

With both external geometry and internal structure defined, the static displacement due to the loading condition computed using the VLM code may then be determined. More important, the natural vibration modes of the structure are calculated as well as the overall structural weight (Fig. 8).

Lastly, the controls for the operation of the optimizer module, defining variables for optimization and their respective ranges, are illustrated in Fig. 9. The optimizer, which controls the activity of the other modules, makes use of a gradient-based algorithm (feasible sequential quadratic programming) to search for the minimum of a user-defined objective function, and can be coupled with both direct-analysis and surrogate-model modules [25]. It replaces a previously implemented MATLAB-based optimizer with the added advantages of being faster and of supporting multithreading. Also, the optimization algorithm returns iterates that satisfy inequality and linear-equality constraints, meaning that if the optimization is interrupted for whatever reason, a design point that respects at least these constraints will be available. Among geometric, aerodynamics, and structural parameters, there is a choice of over 300 different variables to be considered for optimization.

IV. Optimization results

The comparison between the optimization methodologies introduced earlier is made by solving four different case studies, employing the tool described in the previous section. The first three case studies aim at evaluating the performance impact of the number of design variables/complexity of the objective function (note, the problems presented here are not truly scalable). The final case study, in which aerostructural optimization is performed, represents a typical aircraft-design problem.

The baseline planform used is trapezoidal, and a schematic is presented in Fig. 10. The wing section airfoil is a symmetric NACA0009 at the three customizable sections (root, breakstation, and tip). Flight conditions are also the same for all optimization problems: M0.7 at 11,000 m (International Standard Atmosphere), $\alpha = 2$ deg.

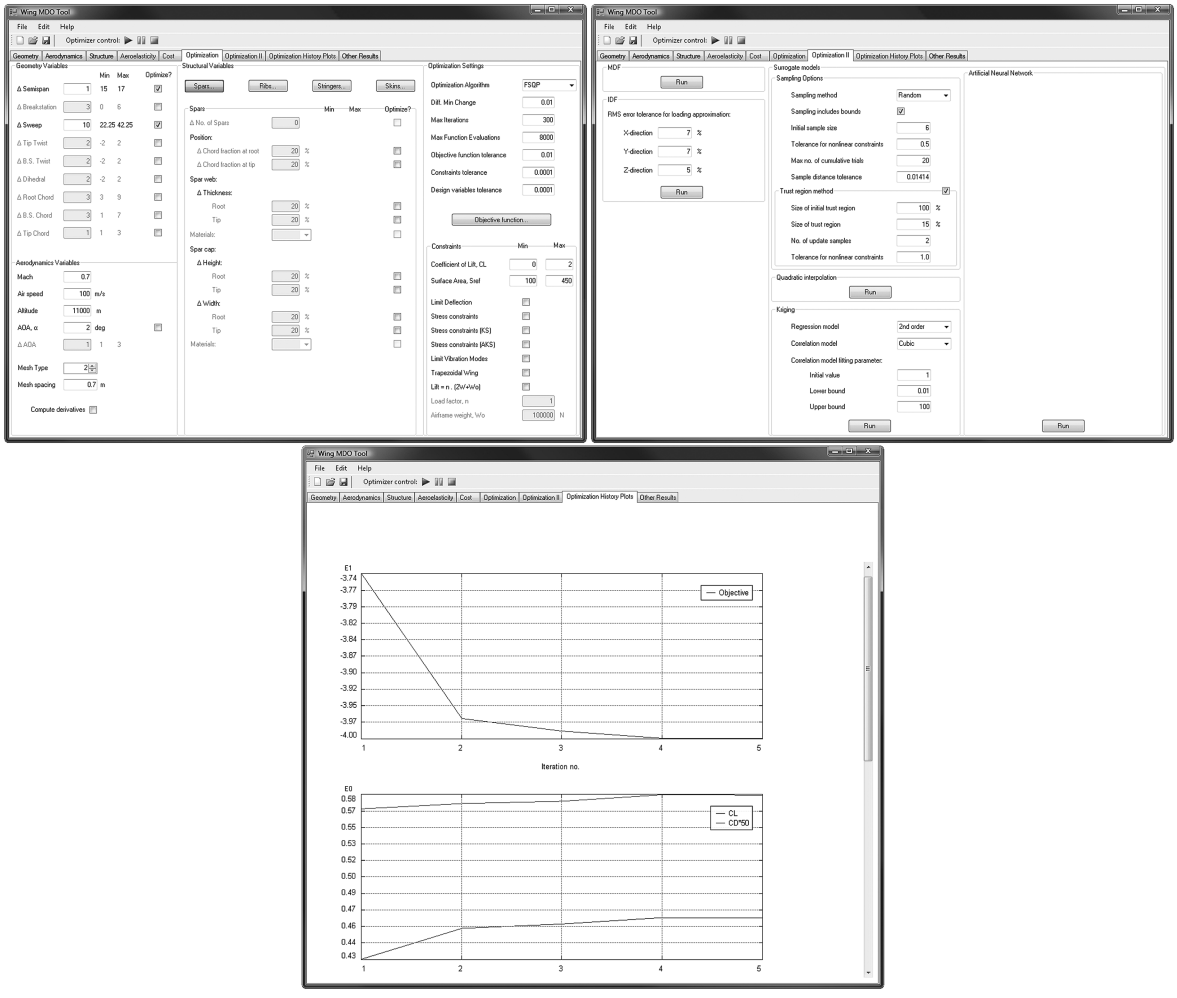

Fig. 9 Optimization tabs.

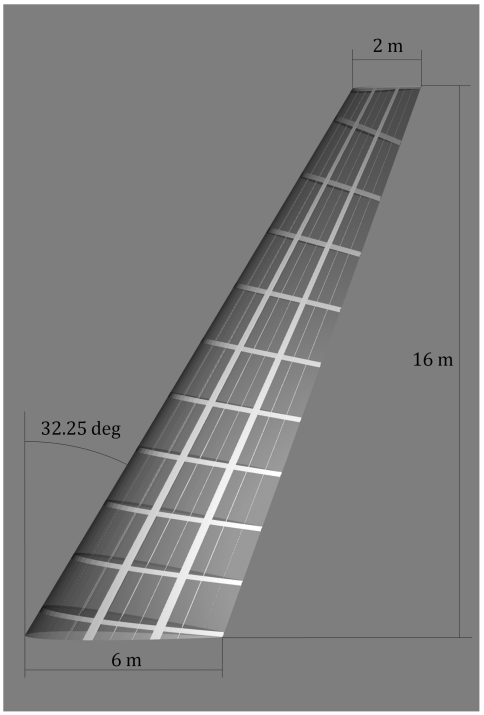

Fig. 10 Baseline wing planform and internal structure.

As mentioned before, the Prandtl–Glauert transformation is applied to the VLM model to account for the effects of compressibility.

The initial wing structure possesses 2 spars, 11 ribs, and 8 pairs of stringers (same position in lower and upper surfaces), built in aluminum, and a carbon-fiber skin. The data does not portray any real wing but is nonetheless a feasible design in terms of structural constraints.

Because of the stochastic nature of the methods being employed (due to the sampling scheme), several trial runs must be executed with each of them. More specifically, for each method, at least 30 independent runs (same starting point) were performed so the sample could be considered statistically significant. In most cases, a normal distribution is then assumed to fit the results and helps in determining whether in a probabilistic sense one method has an advantage over the other, and how well it fares against a conventional non-approximative approach. The comparison is performed using three main criteria: the number of function evaluations (actual analysis and not metamodel evaluations) until convergence, the final objective error (in terms of the percentage of the difference between initial and optimum objective function values), and the configuration error (percent error in the optimum design variables, measured as the norm of the difference in optimum **x** vectors). The last two are measured with respect to the conventional optimization results. The number of function evaluations, while not a continuous random variable, can be considered as such for the purposes of building a normal distribution plot, since the size of the discretization grid is generally small compared with its spread.

In terms of sampling, for both Kriging- and ANN-based methods, the initial sample was limited to $(n_{\mathrm{DV}}+1)(n_{\mathrm{DV}}+2)/2$, whereas the update sample simply added n_{DV} successful samples (in the sense of what is described in Sec. II.D). The trust region for these models is initially set to the whole design space for the initial build, and all subsequent sampling is performed in a hypercube centered on the current iterate and delimited at $\pm 15\%$ of the design variable range.

Table 2 Design variables and constraints for Case Study 1

Design variable	Min	Max
Semispan $b/2$	15 m	17 m
LE sweep Λ_{LE}	22.25 deg	42.25 deg

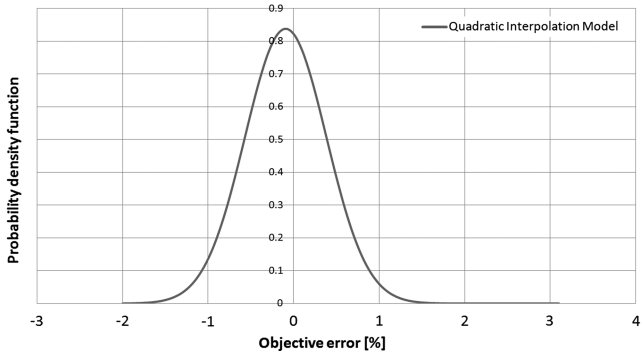

Fig. 11 Distribution of the error in optimum value of the objective function (Case Study 1).

Because the implementations of both Kriging and ANN are highly customizable, a default parameter set that would offer a good compromise between speed and robustness had to be determined. With the Kriging metamodel, a cubic correlation function was employed for all cases, except in Sec. IV.C, where a Gaussian correlation is also used for comparison purposes. A quadratic regression model is in use at all times. Neural networks were built with a single layer containing $4n_{\mathrm{DV}}$ neurons, and the learning rate was set at 0.01.

A. Case Study 1: two design variables

The simplest problem being solved in this series deals with the aerodynamic shape optimization of a wing with respect to its semispan $b/2$ and leading-edge sweep angle Λ_{LE}. The objective, to maximize the lift-to-drag ratio, is shared with Case Study 2 and 3 as well. The bounds for the DVs are presented in Table 2.

The optimum point is, in this case, at $b/2 = 17$ m and $\Lambda_{\mathrm{LE}} = 22.25$ deg (resulting in an L/D ratio of 28.64), and is captured precisely by all of the optimization strategies employed.

In terms of the number of function evaluations required to reach such an optimum, the quadratic response surface is the least costly, consistently converging with only six function evaluations, albeit with a certain amount of error in following the true value of the objective function. This fact is evidenced in Fig. 11, where a normal distribution was assumed for the objective error.

As for the other methods, both Kriging and ANN follow the reference set by the conventional optimization with absolutely no error in either objective or configuration. In terms of the number of function evaluations, they average quite close to the reference (nine evaluations), with expected values of 9.4 and 9.2, respectively (the number of function evaluations alternates between 8 and 10 for these methods).

B. Case Study 2: five design variables

In this second optimization problem, the design variables from before are joined by the chords at the root and tip of the wing, as well as the chord at the breakstation (the breakstation spanwise location y_{bs} is in this case fixed at 8 m from the root, and hence the baseline value for the breakstation chord c_{bs} is 4 m). Table 3 presents the design variables limits and constraints.

Table 3 Design variables and constraints for Case Study 2

Design variable	Min	Max
Semispan $b/2$	15 m	17 m
LE sweep Λ_{LE}	22.25 deg	42.25 deg
Root chord c_{root}	3 m	9 m
Breakstation chord c_{bs}	1 m	7 m
Tip chord c_{tip}	1 m	3 m
S_{ref}[a] $\geq$ 100 m^2		

[a]Represents the total wing surface area of the aircraft.

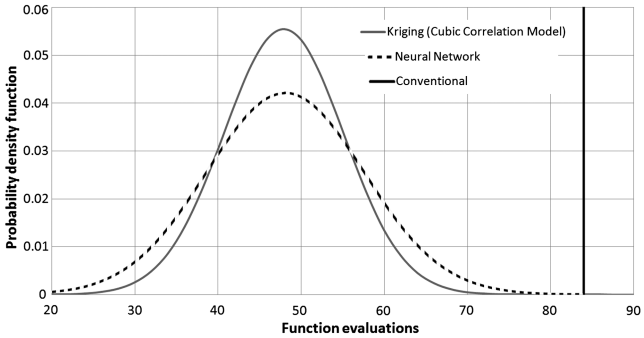

Fig. 12 Distribution of the number of function evaluations (Case Study 2).

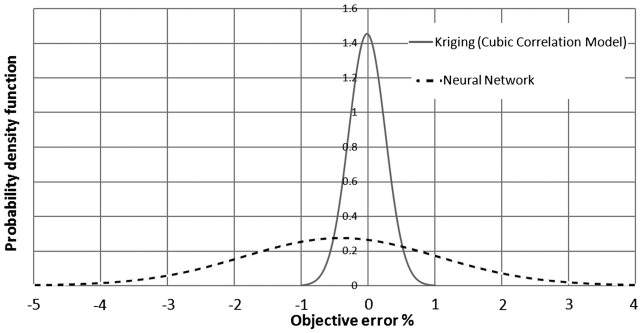

Fig. 13 Distribution of the error in optimum value of the objective function (Case Study 2).

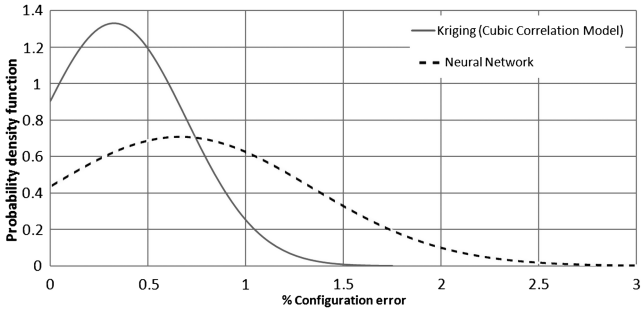

Fig. 14 Distribution of the error in optimum configuration (Case Study 2).

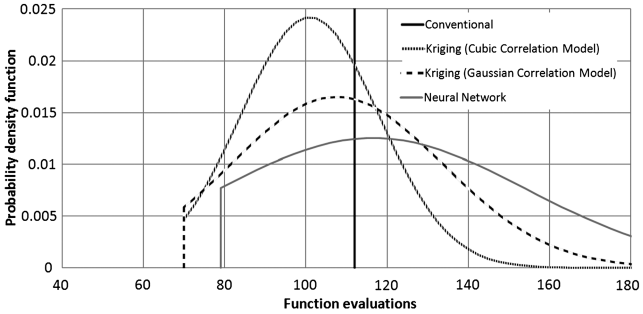

Fig. 15 Distribution of the number of function evaluations (Case Study 3).

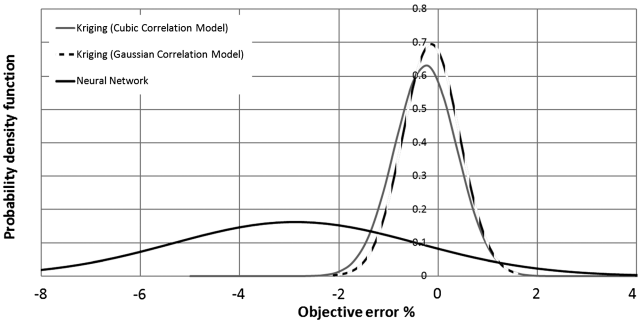

Fig. 16 Distribution of the error in optimum value of the objective function (Case Study 3).

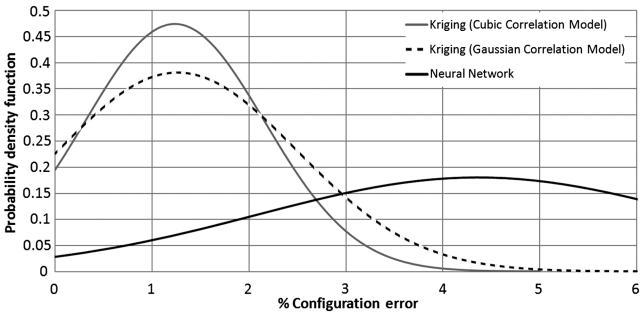

Fig. 17 Distribution of the error in optimum configuration (Case Study 3).

The optimum design configuration was found to be $b/2 = 17$ m, $\Lambda_{\text{LE}} = 22.25$ deg, $c_{\text{root}} = 5.080$ m, $c_{\text{bs}} = 2.962$ m, $c_t = 1$ m. The lift-to-drag ratio is hence maximized to 31.01.

With this number of design variables, the quadratic-interpolation response surface proved unfit for the task and consistently gave poor results (converged to completely different configurations than the optimum, and never the same), for which reason it was henceforth excluded from the comparison. Regarding the remaining methods, the results in Figs. 12–14 indicate that a clear performance advantage over the conventional approach can be attained with minimal cost in objective and configuration error. The benefits lead, on average, to about 50% reduction in the number of function calls both with Kriging and ANN, although the distribution for the latter reveals a higher deviation from the mean.

In these plots, since both the number of function evaluations and the configuration error are, by definition, positive quantities, a truncated normal distribution was employed [lower bound (a) of zero, for the configuration error, or the observed minimum in

Table 4 Design variables and constraints for Case Study 3

Design variable	Min	Max
Breakstation y_{bs}	5 m	11 m
LE sweep Λ_{LE}	22.25 deg	42.25 deg
Breakstation twist θ_{bs}	−2 deg	2 deg
Tip twist θ_{tip}	−2 deg	2 deg
Root chord c_{root}	3 m	9 m
Breakstation chord c_{bs}	1 m	7 m
Tip chord c_{tip}	1 m	3 m
$S_{\text{ref}} \geq 100$ m^2		

Table 5 Design variables and constraints for Case Study 4

Design variable	Min	Max
Semispan $b/2$	15 m	17 m
LE sweep Λ_{LE}	22.25 deg	42.25 deg
Root chord c_{root}	3 m	9 m
Tip chord c_{tip}	1 m	3 m
Skin thickness	0.003 m	0.009 m
Skin-thickness ratio	0.35	1.35
$S_{\text{ref}} \geq 100$ m^2		
$\frac{1}{\rho}\log(\sum_{i=1}^{N} e^{\rho g_i}) \leq 0$, $g_i = f(\sigma_i, \sigma_{\text{yield}})$		

Table 6 Local minima detected for Case Study 4

Design variable	Optimum (MDF)	Optimum 1	Optimum 2	Optimum 3	Optimum 4
Semispan $b/2$	15.028 m	17 m	15.097 m	15.019 m	15.542 m
LE sweep Λ_{LE}	35.31 deg	28.35 deg	40.86 deg	37.02 deg	35.63 deg
Root chord c_{root}	6.161 m	3.708 m	7.977 m	7.402 m	5.370 m
Tip chord c_{tip}	2.238 m	3.003 m	1.000 m	1.000 m	2.995 m
Skin thickness (root)	0.003 m	0.003 m	0.003 m	0.003 m	0.003 m
Skin-thickness ratio	0.35	0.35	0.35	0.35	0.35
$f(\mathbf{x}_{\text{optimum}})$	8.495	8.467	8.483	8.460	8.484
C_D	0.007133	0.007200	0.007005	0.007063	0.007105
m_{wing}	1362 kg	1267 kg	1477 Kg	1397 kg	1379 kg

the case of the number of function evaluations, and upper bound (b) of $+\infty$]:

$$f(x,a,b) = \frac{\frac{1}{\sigma}\phi(\frac{x-\mu}{\sigma})}{\Phi(\frac{b-\mu}{\sigma}) - \Phi(\frac{a-\mu}{\sigma})} = \frac{\frac{1}{\sigma}\phi(\frac{x-\mu}{\sigma})}{1 - \Phi(-\frac{\mu}{\sigma})} \tag{17}$$

where μ is the mean and σ the standard deviation of the fitted distribution, and ϕ and Φ represent the probability density function and cumulative distribution function of the standard normal distribution, respectively.

C. Case Study 3: seven design variables

The final aerodynamic shape optimization problem has a slightly different set of variables, as illustrated by Table 4.

The reference optimum configuration for this problem is: $y_{\text{bs}} = 5$ m, $\Lambda_{\text{LE}} = 22.25$ deg, $\theta_{\text{bs}} = -2$ deg, $\theta_{\text{tip}} = -2$ deg, $c_{\text{root}} =$ 5.675 m, $c_{\text{bs}} = 3.806$ m, $c_t = 1$ m (corresponding to an L/D value of 36.71). As mentioned before, two different correlation models are used with the Kriging model in order to illustrate the effect their choice might have on the results.

A striking difference from the previous case study is the fact that the surrogate-model-based approaches are no longer irrevocably better than conventional optimization (112 function evaluations). For Kriging, the probability of improvement in terms of the number of function calls is now about 75% (obtained from integration of the probability density function up to the reference value) with the cubic correlation model, dropping to around 54% with the Gaussian model (Fig. 15). The ANN approach performs worse than the reference most of the time, since the probability of improvement is now below 50% and shows a tendency to miss the objective by a reasonable amount (objective error on Fig. 16, configuration error on Fig. 17).

It is therefore clear that both surrogate-model types, but especially the ANN approach, would benefit from additional tuning of their respective parameters when solving this particular problem.

D. Case Study 4: aerostructural optimization, six design variables

With aerostructural optimization enabled, four geometry design variables are joined by two structural variables (skin thickness and skin-thickness ratio between root and tip). Constraints are imposed on the maximum admissible stress[§] in each structural component (Kreisselmeier–Steinhauser constraint aggregation is used here to save on the number of nonlinear constraints being imposed), and a trapezoidal planform is enforced. Table 5 summarizes the bounds on design variables and constraints.

Additionally, an equality constraint, simulating a cruise-flight condition, is applied:

$$L - (2 \cdot m_{\text{wing}} g + W_{\text{other}}) = 0 \tag{18}$$

where L is the total wing lift, $m_{\text{wing}} g$ represents the weight of one wing (output variable from the structural module), and W_{other} is the weight of all other aircraft components (fuselage, tail, engines), which is kept constant throughout the optimization process. Note, the either positive or adverse lift of the tail is not being taken into account here, as a first approximation.

Changing from the previous case studies, the objective function is now

$$f(\mathbf{x}) = 1000 C_D + 0.001 m_{\text{wing}} \tag{19}$$

which is weighting both wing drag and structural mass and corrected for the expected magnitudes of each. In this case, the surrogate-model-based optimization approaches are to be compared with a conventional multidisciplinary feasible (MDF) optimization architecture.

For this problem, multiple minima were detected while employing the Kriging approximation. These optima were readily validated using the conventional MDF approach (using as a starting point the average optimum configuration obtained with the Kriging approximation) and are presented in Table 6 as well as in Fig. 18 (compared against the baseline). Beyond the four optima presented here, two others were found, but were not repeated often enough to become statistically significant. The optimum found with the conventional approach is also shown, for comparison purposes (starting point is the usual baseline wing). This effectively proves how this surrogate-model approximation facilitates a more global search of the design space (albeit, independent runs are required). Interestingly, the optimum found through regular MDF was not detected through the Kriging approximation, which instead walks over the latter, yielding yet lower values of the objective function at the minima detected (therefore, the number of function evaluations for the conventional approach, presented in Fig. 19, refers only to the optimization run having the baseline wing as a starting point).

In line with what had been observed in the previous study, the ANN implementation would require more tuning of its parameters since it failed to regularly converge to any of the detected optima and instead frequently converged to suboptimal configurations (but never the same twice). Therefore, it has not been included in the results, as that would require changing critical model parameters, preventing any direct comparison with the previous case studies.

It is duly noted that the design variables that are exclusive to the structural module (skin thickness and skin-thickness ratio between root and tip) are brought to their lower limits by the optimizer in all of the optimum configurations detected. This indicates that there would be room for improvement should these limits be further relaxed.

The results show how much the added complexity of objective and constraints functions in this case affect the conventional approach in terms of number of function evaluations until convergence (depending on the optimum point found). To the Kriging model, however, this added complexity is of little consequence, indicating that the number of design variables plays a more pivotal role.

Regarding the errors in the optimal value of the objective function and optimum configuration, their distribution was assessed for each of the detected optima (and compared with the reference MDF values). The results presented in Figs. 20 and 21 indicate that some optima are usually found with a smaller error relative to the reference than others. For instance, optimum 1 bounces against hard constraints on semispan and leading-edge sweep angle, which helps in reducing the error in these design variables.

[§]For isotropic materials, the Von Mises criterion is used. For composite structures, the Tsai–Hill criterion is employed instead.

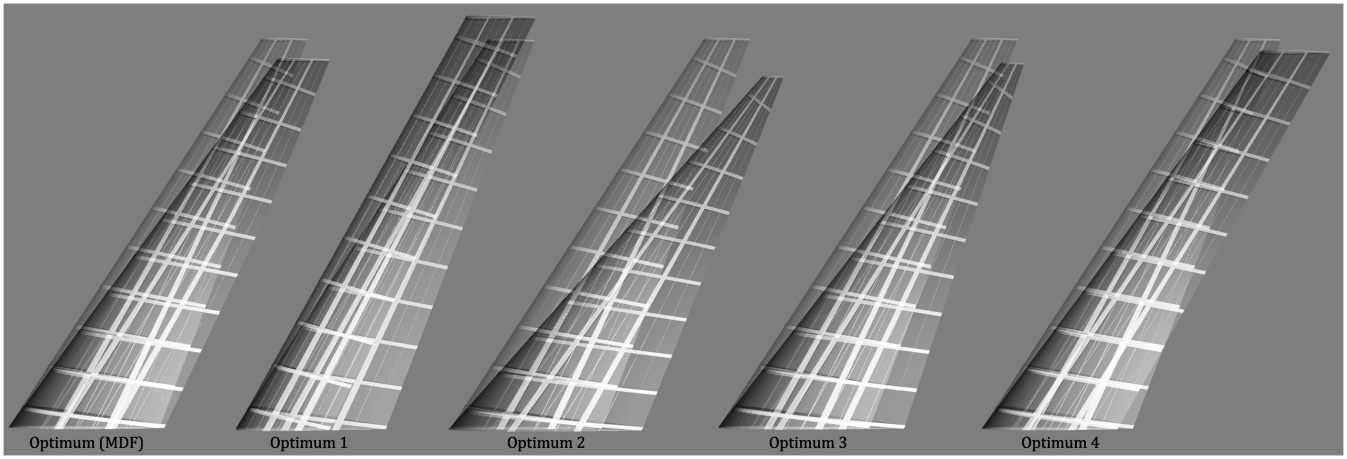

Fig. 18 Optimum configurations (Case Study 4).

Additionally, these optimum points were not detected with the same frequency during the computational experiments. Their frequentist probabilities are: for optimum 1, 26%; for optimum 2, 50%; for optimum 3, 6%; for optimum 4, 18%.

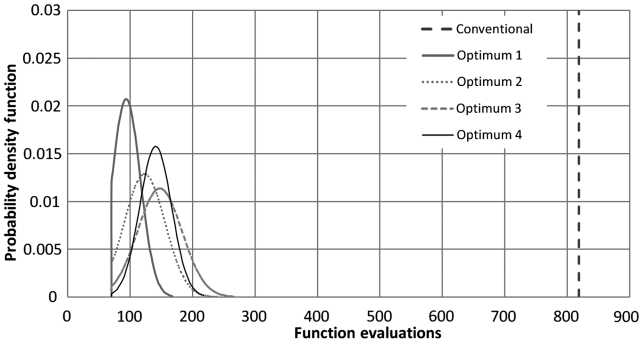

Fig. 19 Distribution of the number of function evaluations (Case Study 4: starting point is baseline wing).

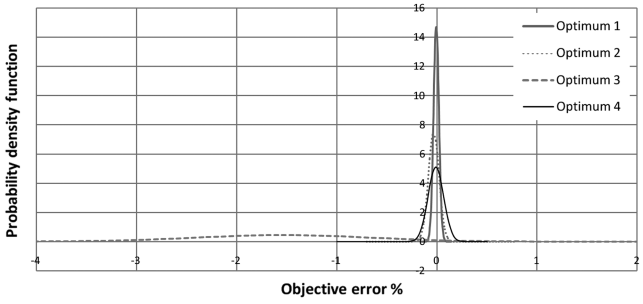

Fig. 20 Distribution of the error in optimum value of the objective function (Case Study 4).

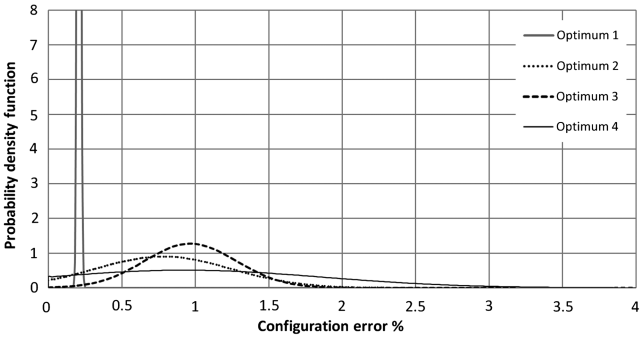

Fig. 21 Distribution of the error in optimal configurations (Case Study 4).

V. Conclusions

First and foremost, it has been concluded that surrogate models are worthy of seeing implementation into design-optimization frameworks such as the one presented here. The performance benefits are difficult to predict, and in some cases knowledge on the underlying physics of the problem would help in better tuning the model (ANN).

A simple adaptive sampling algorithm, coupled with the surrogate-model techniques covered here, was demonstrated to be adequate in directing the optimizer to the reference local optima and, in some cases, to better configurations, frequently with less computational cost. The performance gains become particularly relevant in Case Study 4 (Sec. IV.D), indicating that the number of functions being evaluated (additional, more complex, constraints) has a much larger impact on conventional optimization than on the surrogate-based optimization. It is also a positive fact that this method of successive approximations allows a more thorough exploration of design space.

Nonetheless, residual error in the optimum values of the objective function and design variables may exist at the end of the optimization procedure due to the discrepancies between the surrogate and the actual model. To eliminate it, a hybrid approach could be employed, in which a conventional optimization process is launched using as a start point the optimum found on the surrogate model. Also not covered here was the overhead computational cost of creating the surrogate models, as this was found to be usually negligible compared with the cost of obtaining the sample data.

As mentioned before, these methods are tuned through several parameters (size of trust region, number of samples, etc.), which directly affect their performance outcome. Further investigation is therefore required into which ranges for these parameters are optimal for problems of varying complexity. For instance, with ANN there is no rule of thumb as to how many neurons are required as a function of the dimensionality of the problem, a fact that became apparent in the case studies covered here. The latter helps in explaining why overall Kriging proved to be more robust and better performing than ANN as a surrogate model.

In summary, though the problems used for comparison are not truly scalable, their increasing complexity does have a measurable effect. Only for very low dimensionality problems did quadratic interpolation prove to be adequate. As the number of variables increases, ANN and Kriging start losing their advantage relative to a more conventional approach. Notwithstanding, the Kriging model clearly regains this advantage with the inclusion of nonlinear equality and inequality constraints in the more complex problem dealt with in Case Study 4.

Acknowledgments

This work was supported in part by the Fundação para a Ciência e Tecnologia under Grants SFRH/BD/27863/2006 and SFRH/BD/22861/2005, as well as by Aernnova, Aerospace S.A. The authors

also acknowledge AEM Design for supplying the CFSQP optimization code.

References

[1] Rijpkema, J. J. M., Etman, L. F. P., and Schoofs, A. J. G., "Use of Design Sensitivity Information in Response Surface and Kriging Metamodels," *Optimization and Engineering*, Vol. 2, No. 4, 2001, pp. 469–484. doi:10.1023/A:1016098623669

[2] Giunta, A. A., "Aircraft Multidisciplinary Design Optimization Using Design of Experiments Theory and Response Surface Modeling Methods," Multidisciplinary Analysis and Design Center for Advanced Vehicles, Virginia Polytechnic Institute and State Univ., TR 97-05-01, Blacksburg, VA, May 1997.

[3] Kanazaki, M., Tanaka, K., Jeong, S., and Yamamoto, K., "Multi-Objective Aerodynamic Optimization of Elements' Setting for High-Lift Airfoil Using Kriging Model," *44th Aerospace Sciences Meeting and Exhibit*, AIAA, Reston, VA, 2006.

[4] Kumano, T., Jeong, S., Obayashi, S., Ito, Y., Hatanaka, K., and Morino, H., "Multidisciplinary Design Optimization of Wing Shape for a Small Jet Using Kriging Model," *44th Aerospace Sciences Meeting and Exhibit*, AIAA, Reston, VA, 2006.

[5] Matias, J. M., Vaamonde, A., Taboada, J., and Gonzalez-Manteiga, W., "Comparison of Kriging and Neural Networks With Application to the Exploitation of a Slate Mine," *Mathematical Geology*, Vol. 36, No. 4, 2004, pp. 463–486. doi:10.1023/B:MATG.0000029300.66381.dd

[6] Willmes, L., Baeck, T., Jin, Y., and Sendhoff, B., "Comparing Neural Networks and Kriging for Fitness Approximation in Evolutionary Optimization," *Congress on Evolutionary Computation*, IEEE Publications, Piscataway, NJ, 2003, pp. 663–670.

[7] Paiva, R. M., "Development of a Modular MDO Tool for Preliminary Wing Design," Master's Thesis, University of Victoria, BC, Canada, Dec. 2007.

[8] Hengl, T., "A Practical Guide to Geostatistical Mapping of Environmental Variables," Joint Research Centre, Institute for Environment and Sustainability, TR EUR 22904 EN, Ispra, Italy, Sept. 2007.

[9] Huang, D., "Experimental Planning and Sequential Kriging Optimization Using Variable Fidelity Data," Ph.D. Thesis, Ohio State University, Columbus, OH, 2005.

[10] Lee, K.-H., and Park, G.-J., "A Global Robust Optimization Using Kriging Based Approximation Model," *JSME International Journal*, Vol. 49, No. 3, 2006, pp. 779–788. doi:10.1299/jsmec.49.779

[11] Simpson, T. W., Mauery, T. M., Korte, J. J., and Mistree, F., "Kriging Models for Global Approximation in Simulation-Based Multidisciplinary Design Optimization," *AIAA Journal*, Vol. 39, No. 12, 2001, pp. 2233–2241. doi:10.2514/2.1234

[12] Lophaven, S. N., Nielsen, H. B., and Søndergaard, J., "DACE: A MATLAB Kriging Toolbox," Informatics and Mathematical Modelling, Technical University of Denmark, TR IMM-TR-2002-12, DK-2800 Kgs. Lyngby–Denmark, Aug. 2002.

[13] Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P., "Design and Analysis of Computer Experiments," *Statistical Science*, Vol. 4, No. 4, 1989, pp. 409–423. doi:10.1214/ss/1177012413

[14] McCulloch, W. S., and Pitts, W., "A Logical Calculus of the Ideas Immanent in Nervous Activity," *Bulletin of Mathematical Biology*, Vol. 52, Nos. 1–2, 1990, pp. 99–115.

[15] Suykens, J. A. K., Vandewalle, J. P. L., and Moor, B. L. R. D., *Artificial Neural Networks for Modelling and Control of Non-Linear Systems*, Kluwer Academic , Norwell, MA, 1996.

[16] Rojas, R., *Neural Networks*, Springler–Verlag, Berlin, 1996.

[17] Nguyen, D. H., and Widrow, B., "Neural Networks for Self-Learning Control Systems," *International Journal of Control*, Vol. 54, No. 6, 1991, pp. 1439–1451. doi:10.1080/00207179108934220

[18] Mason, W. H., "Aerodynamics of 3D Lifting Surfaces Through Vortex Lattice Methods: Applied Computational Aerodynamics Text/Notes," Department of Aerospace and Ocean Engineering, Virginia Polytechnic Institute and State University, Blacksburg, VA, March 1998.

[19] Mason, W. H., "Program Friction: Virginia Tech Aerospace Engineering Aerodynamics and Design Software Collection," Department of Aerospace and Ocean Engineering, Virginia Polytechnic Institute and State University, Blacksburg, VA, Jan. 2006.

[20] Chao, D. D., and van Dam, C. P., "Wing Drag Prediction and Decomposition," *Journal of Aircraft*, Vol. 43, No. 1, 2006, pp. 82–90.

[21] Kapania, R. K., and Liu, Y., "Static and Vibration Analyses of General Wing Structures Using Equivalent-Plate Models," *AIAA Journal*, Vol. 38, No. 7, 2000, pp. 1269–1277. doi:10.2514/2.1098

[22] Giles, G. L., "Equivalent Plate Analysis of Aircraft Wing Box Structures with General Planform Geometry," *Journal of Aircraft*, Vol. 23, No. 11, 1986, pp. 859–864. doi:10.2514/3.45393

[23] Livne, E., "Equivalent Plate Structural Modeling for Wing Shape Optimization Including Transverse Shear," *AIAA Journal*, Vol. 32, No. 6, 1994, pp. 1278–1288. doi:10.2514/3.12130

[24] Livne, E., and Navarro, I., "Nonlinear Equivalent Plate Modeling of Wing-Box Structures," *Journal of Aircraft*, Vol. 36, No. 5, 1999, pp. 851–865. doi:10.2514/2.2519

[25] Lawrence, C. T., Zhou, J. L., and Tits, A. L., "User's Guide for CFSQP Version 2.5: A C Code for Solving (Large Scale) Constrained Nonlinear (Minimax) Optimization Problems, Generating Iterates Satisfying All Inequality Constraints," Electrical Engineering Department and Institute for Systems Research, University of Maryland, College Park, MD, April 1997.

A. Messac
Associate Editor