2nd International Conference on Engineering Optimization
September 6-9, 2010, Lisbon, Portugal

A Robust and Reliability Based Design Optimization Framework for Wing Design

Ricardo M. Paiva, André Carvalho, Curran Crawford

University of Victoria, Victoria, British Columbia, Canada

Luís Felix, Alexandra A. Gomes, and Afzal Suleman

Instituto Superior Técnico, Lisbon, Portugal

Abstract
This paper presents the outline of a framework for simultaneous analysis and robustness and reliability calculations in aircraft design optimization, with the option of employing surrogate models. Robust Design Optimization and Reliability Based Design Optimization are merged into a unified formulation which streamlines the setup of optimization problems and aims at preventing foreseeable implementation issues. The code in development expands upon and, in some cases, completely rewrites a previous version of a Multidisciplinary Design Optimization tool that was solely oriented to deterministic problems.
Keywords: Multidisciplinary Analysis, Robust Optimization, Reliability, Surrogate Models

1. Introduction

The increasing competitiveness in the aerospace industry has manufacturers searching for designs that are robust in the sense that they still perform well in off design conditions (flight conditions or load uncertainty, for example), as well as reliable in the way that they present a low probability of failure. Moreover, in early stages of the design, many parameters are yet unknown or poorly characterized. The classical approach to structural design employing safety factors has frequently proved to be overly conservative, thus leaving room for improvement and achieving an edge over competitors. Reliability Based Design Optimization (RBDO) is therefore aimed at seeking a compromise between reliability and cost (not strictly in the financial sense).
This paper details the advancements made in the development of an MDO tool that is aimed at implementing robust and reliability optimization techniques simultaneously, when solving aircraft design problems.

2. Uncertainty based design optimization

To incorporate uncertainties in design optimization implies solving a suitably modified version of the deterministic design optimization problem. The methodologies to do so are divided into two main groups: Robust Design Optimization (RDO) and RBDO. The differences between RDO and RBDO are not mentioned often enough in the literature [1], and as such, a clarification is in order.
RDO methods aim at optimizing the deterministic response of a system about a mean value - maximizing robust performance while minimizing its sensitivity to random parameters. In this way the impact of design uncertainties is harder to perceive (e.g. definition of standard deviation versus probability of a realization residing inside an interval).
On the other hand, RBDO approaches provide a way of designing while taking into account safety margins. In other words, the optimization can be performed while having a particular risk in mind - target reliability (and/or performance).
While both architectures necessarily require more function evaluations than the equivalent deterministic optimization problem, RDO can be performed on an unconstrained problem while RBDO is by definition performed on constrained problems only.

2.1. Robust Design Optimization

A generic statement for a deterministic optimization problem can be:

$$\begin{array}{lll} \min\limits_{\mathbf{x}} & f(\mathbf{x}) & \\ \text{subject to:} & g_i(\mathbf{x}) \leq 0 & i = 1, ..., n_g \\ & x_k^{LB} \leq x_k \leq x_k^{UB} & k = 1, ..., n_{DV} \end{array} \tag{1}$$

where $\mathbf{x}$ is the vector of design variables (DVs), $\mathbf{x}^{LB}$ and $\mathbf{x}^{UB}$ are the lower and upper bounds on the DVs, respectively. Reformulation of the same problem in an RDO perspective would yield [2]:

$$\begin{array}{lll} \min\limits_{\mu_{\mathbf{x}}} & F(\mu_f(\mathbf{x},\mathbf{r}),\sigma_f(\mathbf{x},\mathbf{r})) & \\ \text{subject to:} & G_i\left(\mu_{g_i}(\mathbf{x},\mathbf{r}),\sigma_{g_i}(\mathbf{x},\mathbf{r})\right)\leq 0 & i=1,...,n_g \\ & P(x_k^{LB}\leq x_k\leq x_k^{UB})\geq P_{bounds} & k=1,...,n_{DV} \end{array} \tag{2}$$

where μ and σ represent the mean and standard deviation of the quantities in the subscript (DVs, objective or constraints). These may be computed as per Eqs. 3 and 4. In this formulation, the otherwise deterministic parameters $\mathbf{r}$ are allowed a random distribution (in a typical design problem these may constitute material properties, for instance), and the design variables $\mathbf{x}$ can now be either deterministic or random. The robust objective and constraints are now functions of the mean and standard deviation of objective and constraints, which in turn depend on the probabilistic distribution of the variables. The bounds on the DVs are now themselves established in terms of their probability of residing inside the preset interval.

$$\mu_f(\mathbf{x}) = \int_{-\infty}^{+\infty} f(\mathbf{t})\, p_{\mathbf{x}}(\mathbf{t})\, d\mathbf{t} \tag{3}$$

$$\sigma_f(\mathbf{x}) = \int_{-\infty}^{+\infty} \left[f(\mathbf{t}) - \mu_f(\mathbf{x})\right]^2 p_{\mathbf{x}}(\mathbf{t})\, d\mathbf{t} \tag{4}$$

here f represents a function of interest and $p_{\mathbf{x}}$ is the joint probability density function. The analytical evaluation of the integrals in Eqs. 3 and 4 is impossible in most practical cases, and for that reason a numerical procedure is required. Among the various techniques that may be used are Monte Carlo methods, the Taylor based Method of Moments [2], the Sigma Point technique, and surrogate models approximating $\mu_f(\mathbf{x})$ and $\sigma_f(\mathbf{x})$ directly [3].

2.2. Reliability Based Design Optimization

An equivalent statement to Eq. 1 in RBDO is [4, 5]:

$$\begin{array}{lll} \min\limits_{\mathbf{x}} & f(\mathbf{x},\mathbf{r}) & \\ \text{subject to:} & g_i^{rc}(\mathbf{x},\mathbf{r})\leq 0 & i=1,...,n_{rc} \\ & g_j^{d}(\mathbf{x})\leq 0 & j=1,...,n_d \\ & x_k^{LB}\leq x_k\leq x_k^{UB} & k=1,...,n_{DV} \end{array} \tag{5}$$

The constraints set is now divided into reliability constraints, g_i^{rc}, and other design constraints, g_j^d (for which a reliability target is not established). Alternatively, the objective function may be defined in terms of the probability of exceeding/not exceeding a certain target, which is to be minimized [1]:

$$\begin{array}{lll} \min\limits_{\mathbf{x}} & P(f(\mathbf{x},\mathbf{r}) - target \geq 0)\ or\ P(target - f(\mathbf{x},\mathbf{r})\geq 0) & \\ \text{subject to:} & g_i^{rc}(\mathbf{x},\mathbf{r})\leq 0 & i=1,...,n_{rc} \\ & g_j^{d}(\mathbf{x})\leq 0 & j=1,...,n_d \\ & x_k^{LB}\leq x_k\leq x_k^{UB} & k=1,...,n_{DV} \end{array} \tag{6}$$

The reliability constraints are of the form:

$$g_i^{rc} = P_{f_i} - P_{allow_i} = P(g_i(\mathbf{x},\mathbf{r})\geq 0) - P_{allow_i} \tag{7}$$

$$P(g_i(\mathbf{x},\mathbf{r})\geq 0) = \int_{g_i(\mathbf{x},\mathbf{r})\geq 0} p_{\mathbf{x},\mathbf{r}}(\mathbf{t})\ d\mathbf{t} \tag{8}$$

effectively ensuring that the probability of the originally deterministic constraint g_i being violated is at the most P_{allow_i} - the allowable probability of failure. Determining the probability of failure, P_{f_i}, requires either sampling (again, Monte Carlo methods) or techniques such as the First Order Reliability Method (FORM) and the Second Order Reliability Method (SORM) [4]. While FORM is widely used in reliability analysis, SORM has seen little practical use as it requires higher order information on the objective function and constraints [1, 4, 5, 6].

2.3. R^{2}BDO

A mix of robust objective/reliable constraints is believed to be better suited than the original RBDO formulation for general purpose optimization, particularly when a performance target cannot be readily established for a probabilistic type of objective. If the initial guess for a target is too far off what an actual design can attain, the probability becomes either too close to zero or to one, slowly varying. As can be seen in Fig. 1, when the expected value of performance of the system (μ_f) exceeds the expectations set by the target by a large amount, the objective function becomes insensitive to change since the probability (accounting for the uncertainty in the design variables) is now being measured at either of the tails of the probability density function (PDF). In practice, this means an optimizer would tend to stop prematurely, before finding a true candidate to local/global minimum.

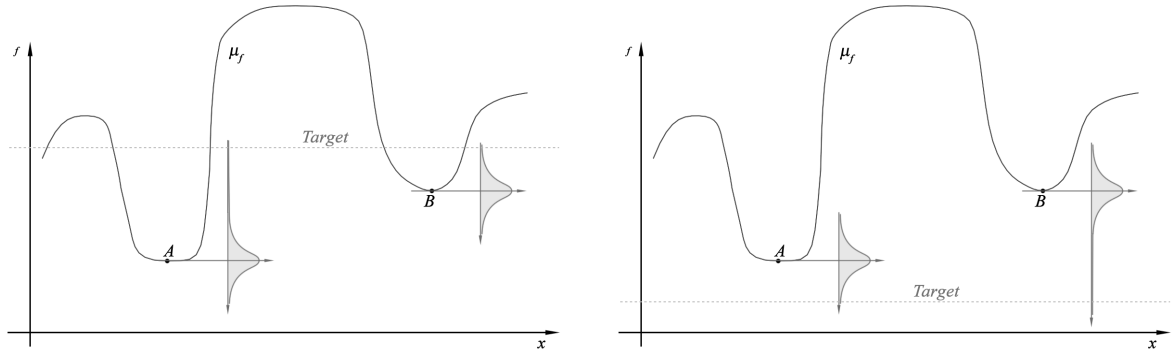

Figure 1: Possible issue with probabilistic objectives

On the other hand, constraint treatment in RDO is not transparent since the designer is left with a choice of weights that will ultimately define how far from the failure surface should the average optimum lie. In [7] and the ubiquitous six-sigma analysis [8], this is addressed by adapting the robust constraints so that the weights are calibrated in order to mimic a probabilistic constraint. This calibration is usually made using the PDF of the normal distribution. Although this is a good approximation for very low input variances or quasi-linear constraints, in general, the constraint output type of distribution may greatly differ from that of the input, thus invalidating this type of analysis. The R^{2}BDO problem is then stated as:

$$\begin{aligned} \min_{\mathbf{x}} \quad & F(\mu_f(\mathbf{x},\mathbf{r}),\sigma_f(\mathbf{x},\mathbf{r})) \\ \text{subject to:} \quad & g_i^{rc}(\mathbf{x},\mathbf{r}) \le 0 \qquad i = 1,...,n_{rc} \\ & g_j^{d}(\mathbf{x}) \le 0 \qquad j = 1,...,n_d \\ & x_k^{LB} \le x_k \le x_k^{UB} \quad k = 1,...,n_{DV} \end{aligned} \tag{9}$$

Essentially, in this proposed hybrid formulation, the type of objective function used in RBDO (either deterministic or probabilistic, as mentioned in the previous section) is replaced by the type used in RDO. In addition, the framework is to make use of surrogate models to expedite the evaluation of objective and constraint functions. For further clarification, a flowchart of the tasks involved in the R^{2}BMDO framework is presented in Fig. 2.

3. MDO Tool

The current version of the MDO tool includes higher fidelity, more capable analyzer modules than previous installments. A three dimensional panel code replaces the vortex lattice method previously in use and a finite element structural analysis code is employed instead of an equivalent plate model [9, 10]. Because user interaction is of prime concern in design optimization, particularly in the initial stages, a graphical user interface (GUI) for the tool was developed. More importantly, 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. 3), where the wing shape 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).

In the aerodynamics module, the results are shown in terms of a pressure distribution contour plot and major aerodynamic coefficients (and their derivatives should the user require it).

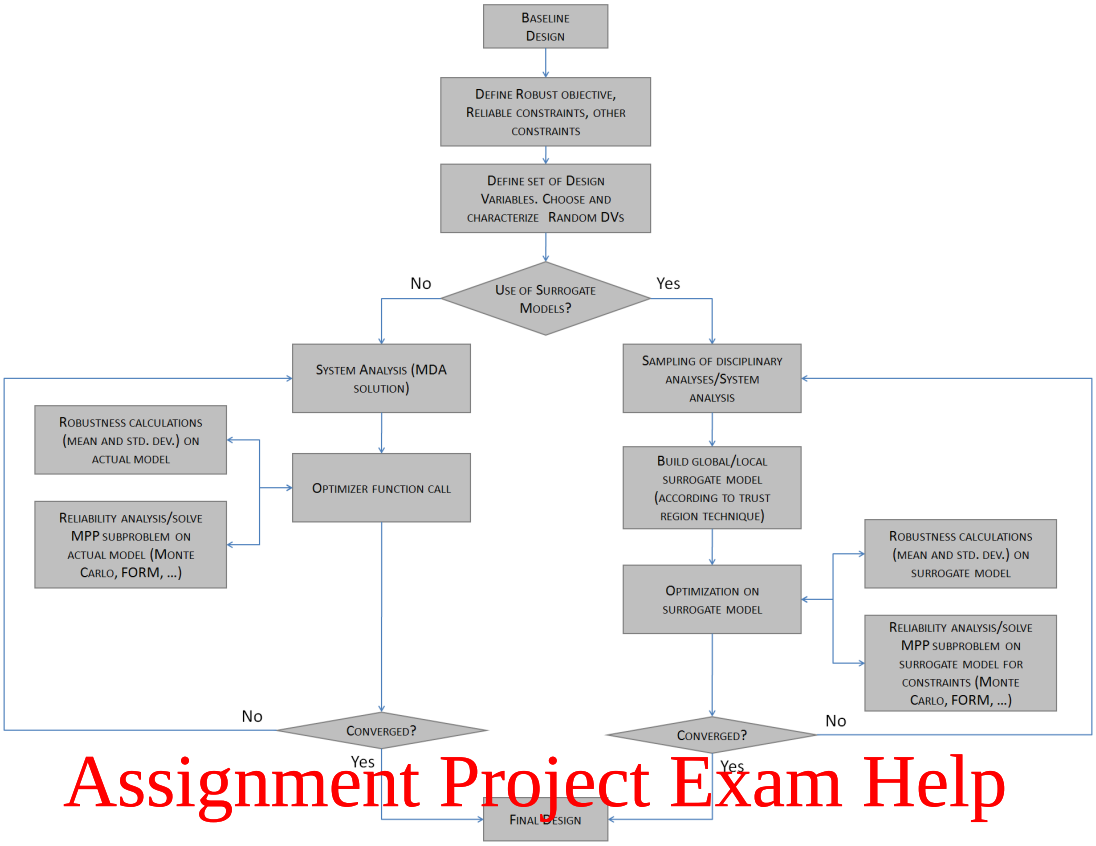

Figure 2: R^{2}BMDO framework layout

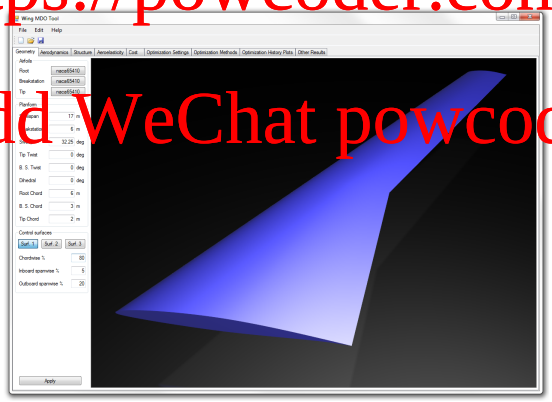

Figure 3: Geometry generation tab.

The doublet/source lattice code in use was developed specifically for this tool and inherits and expands upon the capabilities of the previous aerodynamics module. Contrary to the latter, however, it returns a complete surface pressure field. It also includes compressibility corrections and a skin friction estimate which uses a combination of the Eckert Reference Temperature method for laminar flow and the van Driest II formula for turbulent flow [11]. Support has also been added for the definition of wing mounted pylons and nacelles (with inlet/outlet boundary conditions) as well as control surfaces (see Fig. 4). Inclusion of fuselage and tail assembly is also being considered.

The next module is dedicated to the definition of the wing's structure which remained largely unchanged from previous versions. This is accomplished through a treeview control comprising four main categories: spars, ribs, stringers and skin panels (Fig. 5). Popup menus allow the definition of individual components positioning, section parameters and material properties. Structural analysis is performed through a finite element package which employs University of California FEAP [12] as a solver. The latter was

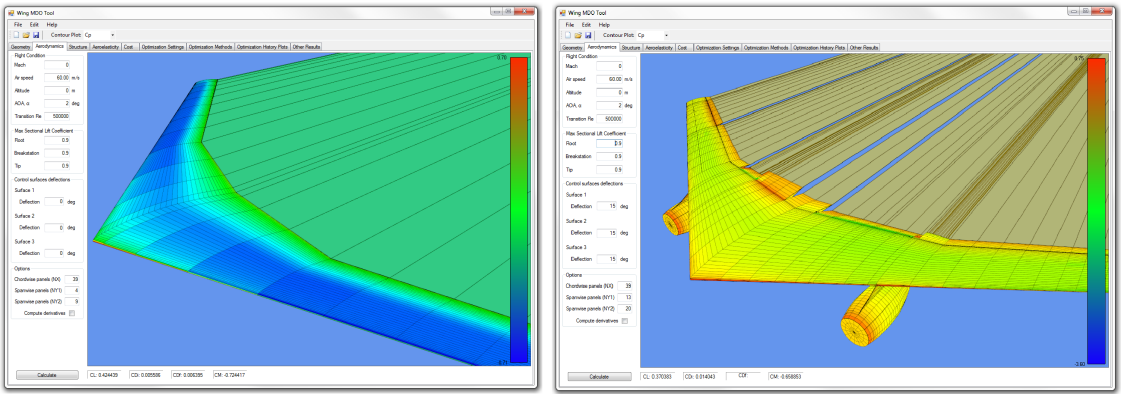

Figure 4: Aerodynamics tab.

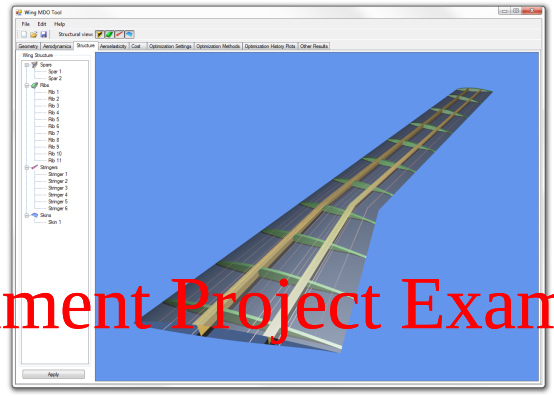

Figure 5: Wing structure definition tab.

coupled with an inhouse developed automatic meshing code which allows the definition of a detailed wing structure composed of spars, ribs, stringers and skin panels (see Fig. 6). The meshing process involves Delaunay triangulation on skin panels, spar and rib caps, and quad-dominant mesh on spar and rib webs. The triangular mesh is required since structural elements may be arbitrarily positioned relative to the planform (linear constraints are however in place to prevent them prevent from overlapping). Stiffners are simply modeled as beam elements. This module is capable of performing both static and modal analysis of the wing structure. Future capability includes provisions for cutouts in the wing skin to accommodate control surfaces and landing gear bay. Lastly, there are the controls for the operation of the optimizer

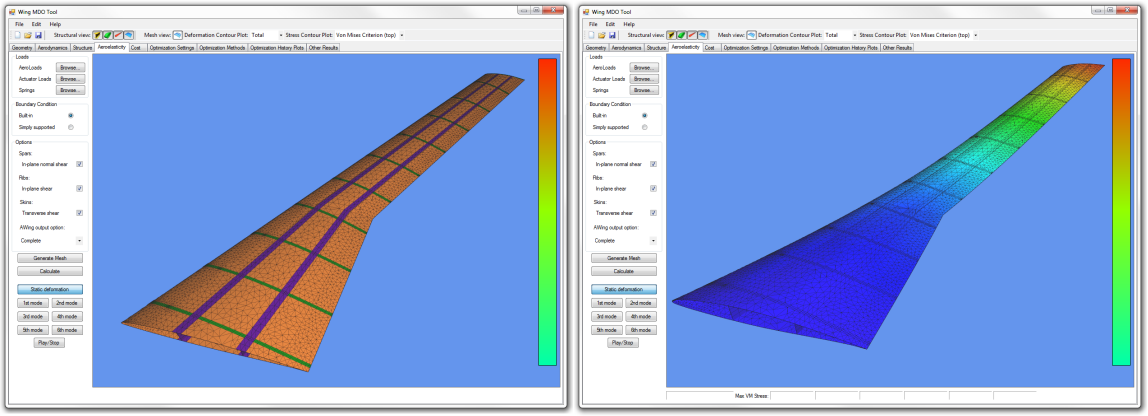

Figure 6: Finite element module.

module, defining variables for optimization and their respective ranges. 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) [13]. It replaces a previously implemented MATLAB based optimizer with the added advantages of being faster and of supporting multi-threading. Among geometric, aerodynamic, and structural parameters, there is a choice of over 300 different variables to be considered for optimization. The optimizer may call the analyzer modules directly or use surrogate models instead. The surrogate model types implemented at present are: quadratic interpolation, regression Kriging and artificial neural networks (these approaches are further elaborated in the following section.

4. Surrogate models

As mentioned before, both RDO and RBDO are computationally much heavier than conventional deterministic optimization due to either the additional design variables that are introduced in the problem or the required evaluation of probabilities. By replacing the computationally expensive analyzers (for aerodynamics, structural analysis, etc.) with suitable surrogate models/metamodels, the optimization process can be greatly accelerated even if at the expense of a loss in fidelity [14]. Much like with RDO and RBDO, in R^{2}BMDO there should also be the possibility of using surrogate models to replace objective and constraint functions, especially in regards to limit state functions and the determination of the MPP. Three response surface/metamodel generating methods have been implemented thus far [10]: quadratic interpolation, Kriging and neural networks. The following sections serve as an introduction to each of them in some detail.

4.1. Quadratic interpolation

Quadratic interpolation based surrogate models are amongst the fastest available (in terms of execution speed at both creation and evaluation time) and their implementation is rather trivial [15]. 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). Although this approach may not be adequate should the interpolated function exhibit oscillatory behavior within its domain, it excels as an inexpensive global model. The interpolating polynomial is therefore written as:

$$\hat{y} = c_0 + \sum_{j=1}^{n_{DV}} c_j x_j + \sum_{k=1}^{n_{DV}} \sum_{j=k}^{n_{DV}} x_k x_j c_{1+n_{DV}+\sum_{m=1}^{k-1}(n_{DV}-m+1)+j-k} \tag{10}$$

The total number of polynomial coefficients (n_t) is in this case equal to the the number of samples to be taken ($n_s = n_t = (n_{DV}+1)(n_{DV}+2)/2$). The fact that sensitivities may be computed analytically is an advantage of this type of 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).

4.2. 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 [16]. Some recent examples of the use of Kriging to develop surrogates to be used in design optimization are the work of Huang [17], Lee and Park, [18] and Simpson et al. [19]. Several types of Kriging have been employed thus far. Some of the most well known variants are Simple Kriging (SK), Ordinary Kriging (OK), and Regression Kriging (RK). 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:

$$y(\mathbf{x}) = f(\mathbf{x}) + z(\mathbf{x}) \tag{11}$$

$f(\mathbf{x})$ 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:

$$f(\mathbf{x}) = \sum_{k=1}^{n_t} \beta_k \, q_k(\mathbf{x}) \tag{12}$$

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 section was not pursued because of diminishing returns of a non locally refined and non-interpolating 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:

$$E\left[z(\mathbf{x})\ z(\mathbf{w})\right] = \sigma^2\,\mathcal{R}\left(\mathbf{w},\mathbf{x},\theta\right) \tag{13}$$

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

$$\hat{y}(\mathbf{x}) = \sum_{k=1}^{n_s} \lambda_k\ y(\mathbf{s}_k) \tag{14}$$

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 which 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 the toolbox [20] as well as in other publications [16, 21]. Derivatives from the Kriging model are also easy to obtain since the derivatives from the regression and correlation functions are well known [20, 16].

4.3. Artificial Neural Networks

Artificial Neural Networks (ANN) were first theorized by Pitts and McCulloch in 1943 [22]. Based on their knowledge of the operation of organic brains, Pitts and McCulloch 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.

The basic unit of ANN is the artificial neuron, which in its simpler form is referred to as a *perceptron*. The *perceptron* is a functional with two components: a weighted summation of the inputs (Eq. 15), and an activation function (Eq. 16) [23, 24].

$$n = W_j x_j - \beta \tag{15}$$

$$y = \gamma\left(n\right) \tag{16}$$

where x_j are the inputs of the *perceptron*, β is a bias and W_j are the weights of each input. γ and y are the activation function and the result of the functional, 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 in data mining [23, 24]. 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 Multi-layer *perceptron* (MLP) is one of the configurations proposed by Pitts and McCulloch and is the most versatile and simple network structure. As the name implies the MLP is comprised of a series of sequentially connected layers of *perceptrons*. A layer is defined as a group of *perceptrons* sharing a common set of inputs and, in the case of the MLP, there cannot be intra-layer connections between *perceptrons* [23].

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, this process starts in the first hidden layer and ends on the output layer.

The mathematical equations of the algorithm are highly dependent on the optimization method being used. For the steepest descent, which is one of the simplest and most common algorithms, Eqs. 17 and 18 represent the first and second parts, respectively:

$$\delta_i^m = \gamma'_{ij}\ W_{jk}^{m+1}\delta_k^{m+1} \tag{17}$$

$$\begin{cases} W_{ij}^m = W_{ij}^m + 2\,\eta\,\delta_i^m x_j^m \\ \beta_i^m = \beta_i^m + 2\,\eta\,\delta_i^m \end{cases} \tag{18}$$

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 the weight matrix and bias vector for layer m 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. In order to be used with the MDO tool, the class was implemented in a dynamic library, thus maximizing portability.

4.4. Sampling

For the purposes of generating the samples required to build the surrogate models, both a random and a Latin Hypercube samplers were implemented. The process of creating the sample space, though customizable by the user, is oriented towards 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. 7, in which the distance to previously taken samples and linear constraints are computed first to avoid a possibly superfluous MDA (Multidisciplinary Analysis) 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. This situation frequently occurs near the intersection of two or more constraints as illustrated in Fig. 8 (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 $g(\mathbf{x}) \leq 0$).

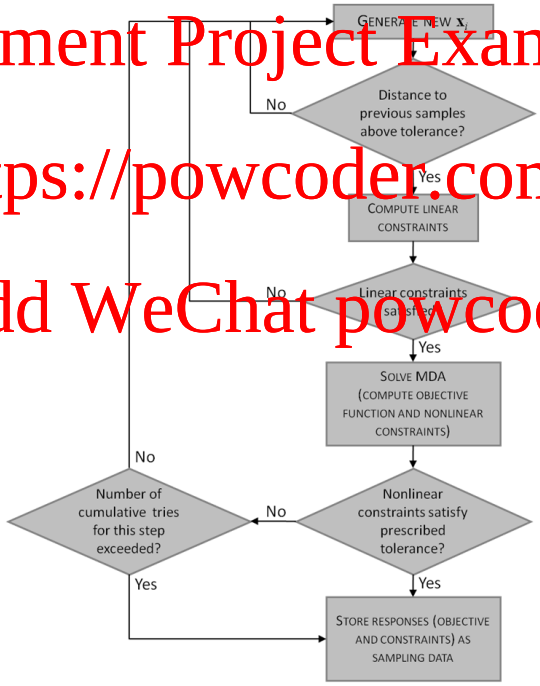

Figure 7: Sampling algorithm with constraint based filtering

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 on Fig. 8.

5. Conclusions

Reliability calculations, in particular, are to require a large number of function evaluations, which will hamper overall performance. The inclusion of surrogate model based optimization practices should then

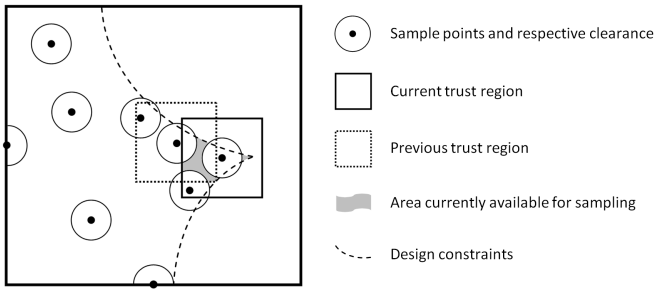

Figure 8: Sampling region in the vicinity of constraints

prove instrumental in the type of framework envisaged. Future work involves the implementation and testing of the proposed R^{2}BMDO architecture, the choice of uncertain parameters and their respective probabilistic distributions, as well as the complete validation of the new analyzer modules. Special attention will be given to assessing the impact of reliability constraints on the results, and determining whether the extra computational effort (vs. deterministic problems) is justified.

Acknowledgements
This work was supported in part by the *Fundação para a Ciência e Tecnologia* (FCT) under Grants SFRH/BD/27863/2006, 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] Allen, M. and Maute, K., "Reliability-Based Shape Optimization of Structures Undergoing Fluid Structure Interaction Phenomena," *Computer Methods in Applied Mechanics and Engineering*, Vol. 194, No. 30-33, 2004, pp. 3472 3495.

[2] Padulo, M., Forth, S. A., and Guenov, M. D., "Robust Aircraft Conceptual Design Using Automatic Differentiation in MATLAB," *Lecture Notes in Computational Science and Engineering*, Vol. 64, 2008, pp. 271 280.

[3] Bonte, M., van den Boogaard, A., and Huétink, J., "Deterministic and Robust Optimisation Strategies for Metal Forming Processes," *Forming Technology Forum 2007 Application of Stochastics and Optimization Methods*, Zurich, Switzerland, 2007.

[4] Agarwal, H., Renaud, J. E., Lee, J. C., and Watson, L. T., "A Unilevel Method for Reliability Based Design Optimization," *45th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference*, AIAA, Palm Springs, California, 2004.

[5] Frangopol, D. M. and Maute, K., "Life-cycle Reliability-Based Optimization of Civil and Aerospace Structures," *Computers and Structures*, Vol. 81, 2003, pp. 397 410.

[6] Frangopol, D. M. and Maute, K., "Reliability-Based Optimization of Civil and Aerospace Structural Systems," *Engineering Design Reliability Handbook*, CRC Press, 2005.

[7] Gerald Steiner, Daniel Watzenig, C. M. and Baumgartner, U., "Statistical Robust Design Using the Unscented Transformation," *The International Journal for Computation and Mathematics in Electrical and Electronic Engineering*, Vol. 24, No. 2, 2005, pp. 606 – 619.

[8] P.N. Koch, R.-J. Y. and Gu, L., "Design for Six Sigma Through Robust Optimization," *Struct Multidisc Optim*, Vol. 26, 2004, pp. 235 248.

[9] Paiva, R. M., *Development of a Modular* MDO *Tool for Preliminary Wing Design*, Master's thesis, University of Victoria, British Columbia, Canada, Dec. 2007.

[10] Paiva, R. M., Carvalho, A. R. D., Crawford, C., and Suleman, A., "Comparison of Surrogate Models in a Multidisciplinary Optimization Framework for Wing Design," *AIAA Journal*, Vol. 48, No. 5, 2010, pp. 995 1006.

[11] Mason, W. H., "Program FRICTION - Virginia Tech Aerospace Engineering Aerodynamics and Design Software Collection," Tech. rep., Department of Aerospace and Ocean Engineering, Virginia Polytechnic Institute and State University, Jan. 2006.

[12] Taylor, R. L., "FEAP - A Finite Element Analysis Program," Tech. rep., Department of Civil and Environmental Engineering, University of California at Berkeley, March 2008.

[13] 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," Tech. rep., Electrical Engineering Department and Institute for Systems Research, University of Maryland, April 1997.

[14] 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, 2002, pp. 469 – 484.

[15] Giunta, A. A., "Aircraft Multidisciplinary Design Optimization Using Design of Experiments Theory and Respons Surface Modeling Methods," Tech. Rep. MAD Center Report 97–05–01, Multidisciplinary Analysis and Design Center for Advanced Vehicles, Virginia Polytechnic Institute & State University, Blacksburg, VA, May 1997.

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

[17] Huang, D., *Experimental Planning and Sequential Kriging Optimization Using Variable Fidelity Data*, Ph.D. thesis, Ohio State University, Ohio, USA, 2005.

[18] 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.

[19] 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.

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

[21] 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.

[22] McCulloch, W. S. and Pitts, W., "A Logical Calculus of the Ideas Immanent in Nervous Activity," *Bulletin of Mathematical Biology*, Vol. 52, No. 1-2, 1990, pp. 99 – 115.

[23] 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 Publishers, 1996.

[24] 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.