Nord Est SUD Ouest
INTERREG IIIC

MATEO

PROJET COFINANCE
PAR L'UNION EUROPEENNE

MATEO SUBPROJECT

ANTASME

WP 4 FINAL REPORT

This document covers deliverables

4.1: New evolution algorithms.
4.2. Validation Report.

LIFT MAXIMIZATION WITH UNCERTAINTIES ON ANGLE OF ATTACK FOR HIGH LIFT DEVICES OPTIMIZATION

Zhili Tang[⋆†] **Jacques Périaux**[†] **Eugenio Oñate**[†] **Gabriel Bugeda**[†]

[⋆] College of Aerospace Engineering, Nanjing University of Aeronautics & Astronautics
29, Yudao Street 210016, Nanjing, China
e-mails: tangzhili@nuaa.edu.cn

[†] Centre Internacional de Métodes Numérics en Enginyeria, Universitat Politécnica Catalunya
Edificio C-1, Campus Norte-UPC, Gran Capitá, s/n, 08034 Barcelona, Sapin

Abstract. *In this report, we address aerodynamic shape optimization problems including uncertain operating conditions. After a review of robust control theory and the possible approaches to take into uncertainty, we propose to use Taguchi robust design methods in order to overcome single point design problems in Aerodynamics. The latter techniques produce solutions that perform well for the selected design point but have poor off-design performance. Under the conduct of Taguchi concept, a design with uncertainties is converted into an optimization problem with two objectives which are mean performance and its variance, so that the solutions are as less insensitive to the uncertainty of the input parameters as possible. Furthermore, the Multi-Criterion Evolutionary Algorithms (MCEAs) are used to capture a set of compromised solutions (Pareto front) between these two objectives. The flow field is analyzed by Navier-Stokes computation. In order to reduce the number of expensive evaluations of fitness function, Response Surface Modelling (RSM) is employed to estimate fitness value using the polynomial approximate model. The proposed approach is applied to the robust optimization of the 2D high lift devices of a business aircraft, by maximizing the mean and minimizing the variance of the lift coefficients with uncertain free-stream angle of attack at landing and takeoff flight conditions respectively*

Key words: Design with Uncertainties, Taguchi Robust Control Methods, Lift Maximization, Evolutionary Algorithms, Semi-Torsional Spring Analogy, Response Surface Modelling

1 INTRODUCTION

The purpose of design engineering is to design better products. This description is intentionally vague, but it suffices to command the attention of the numerical optimizer. Why not design the best products? Why not automatize the design process and replace engineering guesswork with the powerful methods of numerical optimization? In fact, design optimization already has become a vitally important subject of industrial and academic research.

The aircraft design process is generally divided into three stages: conceptual design, preliminary design, and final detailed design. The aerodynamic design plays a leading role during the preliminary design stage where the external aerodynamic shape is typically finalized. The final design would normally be carried out only on the commercially promising completion of the preliminary stage, which makes the preliminary design stage crucial for the overall success of the project.

Optimization of aerospace vehicles or aircraft components is based on a mathematical model of the physical reality. The design variables are parameters in the mathematical model and changes in these design variables result in new physical objects. Aerodynamic optimization is a process of finding a set of design variables that corresponds to a new physical object with better aerodynamic and structural properties.

The aerodynamic design process is embedded in the overall preliminary design with the starting point coming from the conceptual design. The inner loop of aerodynamic analysis is included within an outer (multidisciplinary) loop that is a part of a major design cycle. Because of the limitations of the overall design technology, this cycle is usually repeated a number of times.

Drela (1998, Subsection 5.1)[1] pointed out that if presented with sufficient design model resolution, an optimizer will readily (and annoyingly) manipulate and exploit the flow at the smallest significant physical scales present that tends to produce improved performance only near the sampled operating conditions. The point-optimized airfoil often shows a possibly severe degradation in the *offdesign* performance and optimized aerodynamic shapes are usually *noisy* and usually require a posteriori smoothing.

In the case of aircraft, flow conditions are subject to change during operation. Efficiency and engine noise may be different from the expected values because of manufacturing tolerances and normal wear and tear. Engine components may have a shorter life than expected because of manufacturing tolerances. In spite of the important effect of operating- and manufacturing-uncertainty on the performance and expected life of the component or system, traditional aerodynamic shape optimization has focused on obtaining the best design given a set of deterministic flow conditions. Clearly it is important to both maintain near-optimal performance levels at off-design operating conditions, and ensure that performance does not degrade appreciably when the component shape differs from the optimal shape because of manufacturing tolerances and normal wear and tear. These requirements naturally lead to the idea of robust optimal design wherein the concept of robustness to various perturbations is built into the design optimization procedure.

Recognition of the importance of incorporating the probabilistic nature of the variables involved in designing and operating complex systems has led to several investigations in the recent past. Some of the basic principles of robust optimal design are discussed by Egorov et al.[2]. Several commonly used approaches such as maximizing the mean value of the performance metric, minimizing the deviation of this metric and, maximizing the probability that the efficiency value is no less than a prescribed value are discussed in their paper. Egorov et al[2]. make the observations that a) robust design optimization is in essence multi-objective design optimization because of the presence of the additional objective (robustness) and, b) the addition of the robustness criterion may result in an optimal solution that is substantially different from that obtained without this criterion. Various approaches to robust optimal design are also mentioned in this article.

The main objective of this paper is to develop a robust aerodynamic optimization scheme for achieving consistent lift maximization over a given range of angle of attack with only a few design points. The term *robustness* has been used to mean a variety of things. For optimization under uncertainty, we can distinguish the following meanings and goals of *robust optimization*.

1. Identify designs that minimize the variability of the performance under uncertain operating conditions. This is the objective of Taguchi methods (Fowlkes et al. 1995)[3], which are most practical when the expected value of the performance can be adjusted at negligible cost.
2. Mitigate the detrimental effects of the worst-case performance. This is the objective of $Minimax$ strategies, which can be used to find a design with the optimal worst-case performance (Ben-Tal and Nemirovski 1997)[4].
3. Provide the best overall performance of a system by maximizing the expected value of its utility (Huyse and Lewis 2001)[5][6][7].
4. Achieve consistent improvements of the performance over a given range of uncertainty parameters. This is the main objective of this work.

The imposition of the additional requirement of robustness results in a multiple-objective optimization problem requiring appropriate solution procedures. Typically the costs associated with multiple-objective optimization are substantial. Therefore efficient multiple-objective optimization procedures are crucial to the rapid deployment of the principles of robust design in industry.

Genetic and evolutionary algorithms have been applied to solve numerous problems in engineering design where they have been used primarily as optimization procedures. These methods have an advantage over conventional gradient-based search procedures because they are capable of finding global optima of multi-modal functions (not guaranteed) and searching design spaces with disjoint feasible regions. They are also robust in the presence of noisy data. Another desirable feature of these methods is that they can efficiently use distributed and parallel computing resources since multiple function evaluations (flow simulations in aerodynamics design) can be performed simultaneously and independently on multiple processors. For these reasons genetic and evolutionary algorithms are being used more frequently in design optimization. Deb [8] reviews numerous genetic and evolutionary algorithms for use in multiple-objective optimization. More recent developments such as 1) using hierarchical population topologies, 2) surrogate models to compute the objective function, 3) clustering of data into multiple classes for improved algorithm performance and constructing multiple response surfaces and 4) using non-dominated solutions from previous generations, can be found in Refs.[9][10][11].

Here we focus on applications of an evolutionary algorithm for multiple-objective optimization [12] by using Pareto front concept, which is very useful to the designer because it represents a set which is optimal in the sense that no improvement can be achieved in one objective component that does not lead to degradation at least one of the remaining component. Nowadays EAs benefit from its robustness in capturing convex non-convex discrete or discontinuous Pareto Front of multi-objective optimization problems. In order to capture the non-dominated solution of the Pareto front, EAs work with parent population in the design space as a set of individuals that produces the next generation of children by sorting ranking the set of solution and then building the fitness function using niching techniques.

Applications of this evolutionary method to some difficult model problems involving the complexities mentioned above are also presented in Ref.[12]. The computed Pareto-optimal solutions closely approximate the global Pareto-front and exhibit good solution diversity. Many of these solutions were obtained with relatively small population sizes.

The contribution of computational fluid dynamics (CFD) to aerodynamic design steadily grows due to continuing improvement in the accuracy and robustness of CFD simulations. The past three decades saw a radical change in the entire process of aerodynamic design due to the increasing role of CFD. Until recently, the role of CFD in aerodynamic design was confined to flowanalysis in a limited range of flight conditions and aerodynamic shapes. Additional limitations were caused by the variable accuracy level in the prediction of different aerodynamic characteristics. For example, accurate CFD estimation of drag and pitching moments of three-dimensional shapes became available only in recent years, during which the Navier-Stokes methods reached maturity, therefore Navier-Stokes equations was used to analyze the flow field.

This report deals with uncertainty analysis related to fluctuating operating conditions. We are interested in quantifying the influence of this uncertainty on aerodynamic performance for shape optimization purpose. The final goal of this study is to propose an algorithm to take into account this uncertainty into an automatic shape optimization procedure. We propose to use Taguchi robust design methods in order to overcome single point design problems in Aerodynamics. The latter techniques produce solutions that perform well for the selected design point but have poor off-design performance. Under the conduct of Taguchi concept, a design with uncertainties is converted into an optimization problem with two objectives which are mean performance and its variance, so that the solutions are as less insensitive to the uncertainty of the input parameters as possible. Furthermore, the Multi-Criterion Evolutionary Algorithms (MCEAs) are used to capture a set of compromised solutions (Pareto front) between these two objectives. The flow field is analyzed by Navier-Stokes computation. In order to reduce the number of expensive evaluations of fitness function, Response Surface Modelling (RSM) is employed to estimate fitness value using the approximate model. The propose approach is applied to the robust optimization of the 2D high lift devices of a business aircraft, by maximizing the mean and minimizing the variance of the lift coefficients with uncertain free-stream angle of attack at landing and takeoff flight conditions respectively

The report is organized as follows. In Sect. 2, the optimization problem under uncertainty is defined, then we give a brief introduction to the robust control and design optimizations. Sections 3, a review of the possible approach to take into account uncertainty in the design procedure is achieved. Section 4 is devoted to genetic algorithms. Section 6, response surface approximation modelling is introduced and used in approximating the estimated fitness value. Section 7, semi-torsional spring analogy is used to perform mesh movement according the boundary displacement. The proposed method is demonstrated by optimizing a 2D multi element airfoil of a business aircraft with uncertain of free-stream angle of attack in Sect. 8 and final conclusions are drawn in Sect. 9.

2 ROBUST CONTROL AND DESIGN OPTIMIZATION

2.1 Robust Optimization[13]

Many products are now routinely designed with the aid of computer models. Given the inputs designable engineering parameters and parameters representing manufacturing process conditions the model generates the product's quality characteristics. The quality improvement problem is to choose the designable engineering parameters such that the quality characteristics are uniformly good in the presence of variability in processing conditions.

Let $f : \Re^n \longrightarrow \Re$ be a real-valued function. By *robust optimization* we mean the problem of finding $x^* \in \Re^n$ such that (i) $f(x^*)$ is small, and (ii) x near x^* entails $f(x)$ small. By (ii) we mean more than just the local behavior that any continuous f will necessarily exhibit. Robust optimization is concerned with semi-local behavior and our goal is to avoid simply choosing x^* to be a local minimum of f if that minimizer lies at the bottom of a very steep and narrow basin.

Robust optimization is inherently multi-objective. One way of operationalizing our objectives is to replace $f(x)$ with some measure of f behaves near x—say on some $B(x, r)$, the ball of radius r centered at x. A very conservative possibility is then to measure the performance of f on $B(x, r)$ by its supremum, resulting in the robust optimization problem

$$\min_{x\in\Re^n} \max_{\xi\in B(x,r)} f(x+\xi). \tag{1}$$

This formulation protects against worst-case scenarios and is analogous to the minimax principle in statistical decision theory and elsewhere.

Minimax optimization problems have been the subject of considerable study, e.g. by Dem'yanov and Malozemov (1974)[14], but they are not smooth and are likely too conservative for the present application. A more liberal possibility is to measure the performance of f on $B(x, r)$ by some mean value, so that our objective becomes that of minimizing some sort of moving average of f. This can be accomplished by defining a weighting function (presumably a probability density function) $w : B(0, r) \longrightarrow \Re$, resulting in the robust optimization problem

$$\min_{x\in\Re^n} \int_{B(x,r)} f(x+\xi)w(\xi)d\xi. \tag{2}$$

This formulation addresses typical (as defined by w) scenarios and is analogous to the Bayes principle in statistical decision theory.

Notice that the restriction to $B(x, r)$ is superfluous in problem (2), as it can be incorporated into a specification of a weighting function on $\Re^n$. Furthermore, it is immaterial whether we define the weighting function on $\xi \in \Re^n$ or on $-\xi \in \Re^n$. Hence, the objective function in problem (2) can be written as

$$F(x) = \int_{\Re^n} f(x-\xi)w(\xi)d\xi = [f \otimes w](x) \tag{3}$$

The problem of minimizing F arises in some approaches to global optimization in which one endeavors to smooth the objective function in order to avoid becoming trapped prematurely by a local minimizer. In such methods, however, the weighting function is indexed by a continuation parameter h in such a way that $f \otimes w_h \longrightarrow f$ as $h \longrightarrow 0$. What distinguishes robust optimization is that we are *not* interested in eventually recovering the original objective function.

Both problem (1) and (2) were posed by Das (1996)[15], who preferred the latter because of its smoothness. Das reported that this problem has been studied in the linear, but not the nonlinear programming literature. The major technical difficulty is evidently the evaluation of the n-dimensional integrals that appear in the objective function-especially in light of the fact that n is often quite large in engineering design problems and evaluation f may be quite expensive. As described in Davis and Rabinowita (1984)[**?**] and Flournoy and Tsutakawa (1991)[17], various quadrature and Monte Carlo methods exist for multi-dimensional integration. Unfortunately, the curse of dimensionality guarantees that even highly efficient methods require evaluating the integrand at many points. Accordingly, Das proposed an approximate objective function that is derived by replace $f(x + \xi)$ with its second-order Taylor series expansion at x. Another possibility might be to use Laplace's method, which tends to work well if f is smooth and w is (approximately) normal. This method entails writing the integrand as $\exp(\ln(f(x + \xi)w(\xi)))$, then replacing $\ln(f(x + \xi)w(\xi))$ with its second-order Taylor series expansion at its maximizer. The efficacy of this approach obviously depends on how easily one can determine the maximizer about which to expand; however, laplace's method has proven itself a useful device for calculating posterior expectations in Bayesian statistics, where it has been studied by Tierney and kadane (1986)[18]; Kass, Tierney and Kadane (1989)[19]; and Wong and Li (1992)[20].

2.2 Taguchi and Robust Design[13]

We now turn to the idea of Genichi Taguchi. Taguchi's contributions to quality engineering have been far-ranging and we do not attempt a comprehensive survey in this report. The reader interested in a broader context and a more detailed discussion should turn to the literature on quality control, e.g. Roy (1990)[21], which is the primary source of the material in this section.

Historically, quality engineering relied on product inspection. This remains an important aspect of quality control, but it is evidently a passive approach to developing new and better products. Decades ago, Taguchi proposed that engineers actively design quality into their products, an approach sometimes called *off-line quality control*. Thus, Taguchi was an early proponent of design optimization.

Taguchi envisioned a three-stage process for design optimization. The purpose of the first stage, *systems design*, is to determine the feasible region for the subsequent optimization problem. The purpose of the second stage, *parameter design*, is to optimize the objective function that operationalizes one's notion of quality. This is the stage of particular relevance to the present report. The purpose of the third stage, *tolerance design*, is to fine-tune the (approximately) optimal design obtained in the second stage.

Taguchi had very specific ideas about what objective functions to optimize for parameter design. Perhaps his most fundamental contribution to quality engineering was his observation that one should design a product in such a way as to make its performance insensitive to variation in variables beyond the designer's control. In acknowledgement of the value of this observation, Taguchi is widely regarded as the father of robust design and parameter design is sometimes called robust design.

We now describe some details of Taguchi's strategy for robust design. Taguchi's methods distinguish two types of inputs to a system: *control parameters* (or *control factors*) are the inputs that can be easily controled or manipulated by the designer, hence the inputs that constitute the optimization variables x, *noise variables* (or *noise factors*) are the inputs that are difficult or expensive to control, hence the inputs ξ to whose variation product performance is desired to be insensitive. For example, x might specify the design of (a part of) a photocopier and ξ might specify the environment (temperature, humidity, etc.) in which it is to operate.

To obtain Taguchi's objective function(s), let f denote the quality characteristic of interest. This characteristic may be one of three types, for each of which taguchi defined a measure of mean squared deviation (MSD). Let $f_1, ..., f_N$ denote a sample obtained by varying ξ for a fixed x. If quality is measured by how close f comes to a target value f_m, then

$$MSD = \frac{1}{N}\sum_{i=1}^{N}(f_i - f_m)^2, \tag{4}$$

if quality is measured by how small f is, then

$$MSD = \frac{1}{N}\sum_{i=1}^{N} f_i^2, \tag{5}$$

if quality is measured by how large f is, then

$$MSD = \frac{1}{N}\sum_{i=1}^{N} f_i^{-2}, \tag{6}$$

The objective function to be minimized by varying x is then the *signal to noise ratio* (SNR), $-10\log_{10}(MSD)$. Strong emphasis of these SNRs as the objective functions that operationalize the concerns of robust design is a hallmark feature of Taguchi's methods.

Taguchi's approach to optimizing the SNR was inspired by principles of classical experimental design. The control parameters x are systematically varied according to an orthogonal array, the *control array* or *inner array*. At each value of x, the noise variables are systematically varied according to a second orthogonal array, the *noise array* or *outer array*. At each value of x, data from the *replications* of the quality characteristic f across noise array are used to estimate the SNR. One obtains an array of estimated SNRs, which is then analyzed by standard analysis of variance techniques to identify values of x that produce robust performance.

At this point it is crucial to note that, while Taguchi contributions to the *philosophy* of robust design are almost unanimously considered of fundamental importance, the efficacy of his technique methods for implementing his philosophy is extremely controversial. An excellent survey of these controversial is the panel discussion edited by Nair (1992)[22], in this report we focus on one very specific objection that is particularly damning from the perspective of numerical optimization.

Whatever the details of the SNRs, the control arrays, the noise arrays, the analysis of variance, etc., it is inescapable that taguchi's methods attempt to optimize an objective function by specifying all of the values of x at which the objective function will be evaluated *prior to observing any function values.* (Data analysis is sometimes supplemented by performing one or more confirmatory experiments, but this is not a fundamental part of the optimization strategy.) Thus, the taguchi approach violates a fundamental tenet of numerical optimization-that one should avoid doing too much work until nears a solution. In Taguchi defense, it should be noted that he was the primarily concerned with situations in which sequential experimentation may not be possible, as when an entire manufacturing facility must be dedicated to performing the experiment. In modern engineering design optimization, however, performance is often assessed by computer simulation and the logistic necessity of one-shot experiments disappears.

2.3 Decision-Theoretic Formulations of Robust Design[23]

Using ideas from statistical decision theory, we now describe how the problem of robust design can be formulated as an optimization problem. We consider objective functions of the form $f : X \otimes B \longrightarrow \Re$, where
$x \in X$ represents decision variables, inputs (designs) controlled by the engineer.
$b \in B$ represents uncertainty, inputs not controlled by the engineer.
$f(x, b)$ quantifies the loss that accrues from design x when conditions b obtain.

Our (unattainable) goal is to find $x^* \in X$ such that, for every $b \in B$,

$$f(x^*, b) \leq f(x, b) \quad \forall x \in X \tag{7}$$

Example: Airfoil Shape Design. Suppose that f is the drag coefficient of an airfoil, whose shape is specified by x. Then b might specify:
1. Manufacturing errors ε that perturb the design. The desired airfoil shape is x, but the manufactured airfoil shape is $x - \varepsilon$. The problem is to find a design that will minimize the drag of the manufactured airfoils. This is the problem studied by Welch and Sacks (1991)[24].
2. Mach numbers $M \in [M_b - \varepsilon, M_b + \varepsilon]$, angle of attack $\alpha \in [\alpha_b - \epsilon, \alpha_b + \epsilon]$. The problem is to design an airfoil that performs well over a range of different Mach numbers and angles of attacks. This problem has been considered by several researchers at NASA; the approach pursued by Huyse and Lewis (2001)[6]; Huyse (2001)[7] and Tang (2005)[27] is especially germane to ours.

The unsolvable problem of finding $x^* \in X$ that simultaneously minimizes $f(x, b)$ for each $b \in B$ is the central problem of statistical decision theory: and a decision rule that simultaneously minimizes risk for every possible state of nature. A standard way (e.g., Ferguson, 1967[25]) of negotiating this problem is to replace each $f(x, \cdot)$ with a real-valued attribute of it, e.g.,

Minimax Principle :

$$\min_{x \in X} \phi(x), \quad where \;\; \phi(x) = \sup_{b \in B} f(x; b). \tag{8}$$

Bayes Principle

$$\min_{x\in X}\phi(x), \quad where\ \phi(x)=\int_B f(x,b)p(b)db \tag{9}$$

where p denotes a probability density function on B.

The *minimax* principle is extremely conservative. It seeks to protect the decision-maker against the worst-case scenario. We will adopt the Bayes principle, which seeks to minimize average loss in a sense that can be customized (via the choice of p) to the application. This formulation of the quality control problem was first proposed by Welch, Yu, Kang, and Sacks (1990)[26], although their suggestion appears to have had little effect on engineering practice.

Let $x \in X \subset \Re^n$ denote the design.
1. Let $b = \varepsilon \in B \subset \Re^n$ denote the manufacturing error and let $f(x,b) = f(x-\varepsilon)$ denote the drag coefficient of the manufactured airfoil. We want to minimize

$$\phi(x)=\int_{\Re^n} f(x-\varepsilon)p(\varepsilon)d\varepsilon=[f\otimes p](x) \tag{10}$$

In this case, p might be chosen to approximate the probability distribution of the random manufacturing errors that perturb the design.
2. Let $b \in B = [M_b-\varepsilon, M_b+\varepsilon]\otimes[\alpha_b-\epsilon, \alpha_b+\epsilon]$ denote the Mach number and let $f(x,M,\alpha)$ denote the drag coefficient at Mach M and angle of attack α. We want to minimize

$$\phi(x)=\int_{\alpha_b-\epsilon}^{\alpha_b+\epsilon}\int_{M_b-\varepsilon}^{M_b+\varepsilon} f(x;M,\alpha)p(M,\alpha)dMd\alpha \tag{11}$$

In this case, p is a weight function that quantifies the value placed on performance at different speeds. This problem was studied by Huyse and Lewis (2001)[6] ; Huyse (2001)[7] and Tang (2005)[27].

3 METHODS FOR ROBUST DESIGN[28]

3.1 Multi-Point Approach

In aerodynamics, first attempt to reduce the sensitivity of the optimal design to operating conditions consisted in solving multipoint optimization problems, to prevent from a possible severe degradation in the off-design performance. A multipoint optimization problem consists in replacing the cost function , that is evaluated for a particular set of operating conditions b, by a weighted sum of N cost functions evaluated at different sets of operating conditions $(b_i), i = 1, ..., N$:

$$Minimize\sum_{i=1}^{N} w_i f_i(x,b_i),\ \ x\in\Re^n,\ \ \ b_i\in B,\ \ \ \sum_{i=1}^{N} w_i=1 \tag{12}$$

This strategy has been applied to airfoil design in order to reduce the drag over a Mach range or optimize the shape at cruise and landing flight conditions simultaneously. The main difficulty related to this approach is the well known *point optimization effect*: the solution depends critically on the choice of operating conditions $(b_i), i = 1, ..., N$. Since no requirement is imposed for the cost function at their operating conditions, the optimizer yield a design fully adapted to the N operating conditions under consideration $(b_i)_{i=1,...,N}$, and the optimzed design presents performance degradation for intermediate operating conditions. It was shown in [7] that in order to avoid the point-optimization effect, the number of N of operating conditions should be larger than their number n of design variables. Therefor, this approach cannot be considered as a satisfactory answer to robust design problems.

Some improvement of the classical multipoint approach have been reported in [6] and [7]. In the former, a random choice of the operating conditions $(b_i)_{i=1,...,N}$ in a given interval $[b_{min}, b_{max}]$ for each cost function evaluation was proposed to avoid the tedious choice of operating conditions. In the latter, a new scheme was proposed, called *profile optimization method*, that relies on adaptive adjustments of the weights $(w_i)_{i=1,...,N}$ to achieve a consistent reduction of the cost function f over a range of operating conditions. In [29] adaptive weights is also proposed in conjunction with a new scheme that permits to automatically add new points to prevent from off-design loss.

3.2 Worst-Case Approach

This approach proposed to mitigate the detrimental effects of the worst-case performance. The objective of such a method is to determine a shape, whose worst performance for operating conditions varying in a given interval, is as good as possible. Then, the aim is not to obtain a satisfactory fitness for random fluctuations of operating conditions, but to ensure that the performance degradation at the worst operating conditions is as small as possible.

For a given shape defined by x, suppose that $\tilde{b}$ represents the operating conditions for which the cost function is the worst over a given range $[b_{min}, b_{max}]$:

$$\max_b f(x,b), \quad b \in [b_{min}, b_{max}] \tag{13}$$

The corresponding cost function value is:

$$f_{max}(x) = f(x, \tilde{b}). \tag{14}$$

Then, the shape optimization problems becomes:

$$\min_{x\in\Re^n} f_{max}(x) \tag{15}$$

This problem belongs to the class of well known *Minimax* problems, since for each step of the minimization problem (15), the maximization subproblem (13) has to be solved first. Due to the difficulty to solve such problems (that can be non-differentiable), a discrete formulation is to often adopted in practice by replacing (13) by:

$$\min_{x\in\Re^n} (Maximum\ \ of\ \ f_i(x,b_i),\ \ where\ \ i = 1,...,N) \tag{16}$$

where $(b_i)_{i=1,...,N}$ is a discretization of the interval $[b_{min}, b_{max}]$. Although this formulation seems different than the multipoint approach, it has been shown in the context of airfoil design (see [7]) that these problems can be mathematically equivalent under some assumptions. Therefore, the discrete approach (16) tends also to produce results that depend critically on the choice of operating conditions $(b_i)_{i=1,...,N}$. More generally, this approach only improves the worst-case performance and does not consider the global behavior of the fitness when operating conditions fluctuate.

3.3 Interval Analysis Approach

The basic idea of interval analysis is to determine the interval in which the cost function varies as the operating conditions fluctuate in a given interval. Then, a two-objective optimization problem is solved, that consists in minimizing the median value of the cost function, while minimizing the extent of its interval of variation.

Suppose that operating conditions fluctuate in the interval $[b_{min}, b_{max}]$. For a given shape defined by x, the operating conditions $\hat{b}$ and $\tilde{b}$ that respectively correspond to the best and worst function values, are obtained by solving the two following subproblems:
find $\hat{b}$, which is the solution of optimization problem

$$\max_{b} f(x, b), \quad b \in [b_{min}, b_{max}], \tag{17}$$

find $\tilde{b}$, which is the solution of optimization problem

$$\min_{b} f(x, b), \quad b \in [b_{min}, b_{max}]. \tag{18}$$

The corresponding best and worst cost function values are:

$$f_{min}(x) = f(x, \hat{b}) \qquad\qquad f_{max} = f(x, \tilde{b}). \tag{19}$$

Then, the two-objective shape optimization problem is solved:

$$Minimze \begin{cases} \left(f_{max}(x) + f_{min}(x)\right)/2 \\ \left(f_{max}(x) - f_{min}(x)\right) \end{cases} \quad x \in \Re^n. \tag{20}$$

Contrary to the previous methods, the interval analysis approach clearly defines robust design as a compromise between two goals: increase the fitness of the system and reduce the fitness variations. Despite this improvement, the method has several drawbacks. First, the two subproblem are similar to (13) for the worst-case approach. As explained above, solving directly this kind of problems is tedious for complex applications. Some softwares exists, that automatically perform interval analysis. However, as explained in [30], their use in the context of CFD solvers exhibits significant difficulties, due to the algorithmic complexity of such codes. Secondly, this approach does not take into account the fitness variations inside the interval $[f_{min}(x), f_{max}(x)]$. Only extreme events are considered for the optimization problem.

3.4 Taguchi Approach for Quality Engineering

Taguchi envisioned a three-stage process whereby engineers could design quality into their products. The first stage, systems design, determines the feasible region for subsequent optimization. The second stage, parameter design or robust design, optimizes the objective function that operationalizes one's notion of quality. This is the stage of particular relevance to the present paper. The third stage, tolerance design, fine-tunes the (approximately) optimal design obtained in the second stage.

Taguchi argued that one should design a product in such a way as to make its performance insensitive to variation in variables beyond the designer's control. His methods for robust design distinguish two types of inputs to a system: *control parameters* (or *control factors*) are the inputs that can be easily controlled or manipulated by the designer, hence the inputs that constitute optimization variables x; *noise variables* (or *noise factors*) are the inputs that are difficult or expensive to control, hence the inputs b to whose variation product performance is desired to be insensitive. For example, x might specify the design of a photocopier and b might specify the environment in which it must operate.

Although Taguchi's philosophical contributions to robust design are of fundamental importance, the efficacy of his methods is controversial. These controversies were surveyed in the panel discussion edited by Nair (1992)[22]; here we focus on a specific objection to his method of optimization. This method was inspired by classical experimental design. The control parameters x are systematically varied according to an orthogonal array, the *control array* or *inner array.* At each value of x, the noise variables are varied according to a second orthogonal array, the *noise array* or *outer array*, and data from the *replications* of the quality characteristic across the noise array are used to estimate a signal-to-noise ratio (SNR). One obtains an array of estimated SNRs, which is then analyzed by standard analysis of variance techniques to identify values of x that produce robust performance.

Thus, Taguchi methods attempt to optimize an objective by specifying *a priori* all of the values of x at which the objective will be evaluated. (Data analysis is sometimes supplemented by performing one or more confirmatory experiments, but this is not a fundamental part of the optimization strategy.) Thus, Taguchi approach violates a fundamental tenet of numerical optimization—that one should avoid doing too much work until one nears a solution. In his defense, Taguchi was concerned primarily with situations in which sequential experimentation may not be possible, as when an entire manufacturing facility must be dedicated to performing the experiment. In modern engineering design optimization, however, performance is often assessed by computer simulation and the logistic necessity of one-shot experiments disappears.

In the statistical approach, one consider the fluctuating operating conditions $\mathbf{b} = (b_i)_{i=1,\ldots,N}$ as samples of random variables $\mathbf{B} = (B_i)_{i=1,\ldots,N}$, whose statistical characteristics are known (mean $\mu_{\mathbf{B}} = (\mu^i_B)_{i=1,\ldots,N}$, variance $\sigma^{\mathbf{2}}_{\mathbf{B}} = ({\sigma^2_B}^i)_{i=1,\ldots,N}$, etc). One also suppose for the sake of simplicity that the random variables $(B_i)_{i=1,\ldots,N}$ are independent. the statistical characteristics of operating conditions can be determinied by experimental measurements or engineering experience. Gaussian Probability Density Functions (PDFs) or truncated Gaussian PDFs are often used in practice (see [5] for instance).

The main consequence of this assumption is that the cost function of the problem is also a random variable f. According to the Von Neumann-Morgenstern decision theory, the best choice is then to select the design which leads to the best expected fitness. this is known as the Maximum Expected Values (MEV) criterion. The decision or design that minimizes the risk is known as the *Bayes' decision* and is solution of the following problems:

$$Minimize \;\; \mu_f = \int_{\Omega(B)} f(x,b)\rho_B(b)db, \quad x \in \Re^n, \tag{21}$$

$\Omega(B)$ and ρ_B are the range and the PDF of the random variable B. Then, the MEV criterion just consists in minimizing the statistical mean μ_f of the cost function.

This approach is a significant improvement over previous methods. The robust design problem is now considered with a rigorous statistical framework. this allows to take into account the random fluctuations of the fitness in the optimization problem, but also to care about the frequency of the occurrence of the events, thanks to PDFs. Then, the most probable events have a larger influence in the decision than extreme and unlikely events.

However, problem (21) does not address the variability of the fitness. The mean value of the fitness is the only criterion that is considered in the *Bayes' decision*. For engineering problems, one also would like to select a design for which the fitness is not subject to large variation as operating conditions fluctuate. then, a second criterion is often joined to the MEV criterion, that relies on the minimization of the variance σ_f^2 of the fitness:

$$Minimize \begin{cases} \mu_f = \int_{\Omega(B)} f(x,b)\rho_B(b)db \\ \sigma_f^2 = \int_{\Omega(B)} (f(x,b)-\mu_f)^2\rho_B(b)db \end{cases} \quad X \in \Re^n, \tag{22}$$

This approach aims at determining a trade-off between the expected fitness and the expected fitness variation as operating conditions randomly fluctuate. Although this approach is satisfactory from theoretical and practical viewpoints, its application is not straightforward. Particularly, the estimation of the mean and variance can be tedious from complex CFD applications. This issue is detailed below.

3.4.1 Deterministic viewpoint

For a given design determined by x, the statistical mean μ_f and variance σ_f^2 are defined by:

$$\begin{aligned} \mu_f &= \int_{\Omega(B)} f(x,b)\rho_B(b)db, \\ \sigma_f^2 &= \int_{\Omega(B)} (f(x,b)-\mu_f)^2\rho_B(b)db. \end{aligned} \tag{23}$$

These integrals can not be analytically evaluated since no analytical formulation of f is available. To estimate them, one should either provide an analytical approximation of the cost function or discretize the integrals. The latter solution is quite similar to the multipoint approach, since the integrals become weighted sums of the cost functions evaluated for some prescribed operating conditions. Then, the former approach using an approximation $\tilde{f}$ is usually preferred:

$$\begin{aligned} \tilde{\mu}_f &= \int_{\Omega(B)} \tilde{f}(x,b)\rho_B(b)db, \\ \tilde{\sigma}_f^2 &= \int_{\Omega(B)} (\tilde{f}(x,b)-\tilde{\mu}_f)^2\rho_B(b)db. \end{aligned} \tag{24}$$

3.4.2 Stochastic viewpoint

To estimate the mean and variance of the random variable f, one can simply use statistical estimators in a classical Monte-Carlo approach. A sample of operating conditions $(b_i)_{i=1,\dots,N}$ if size N is generated according to the PDF ρ_B. Then, unbiased estimators of the mean and variance are:

$$\begin{aligned} \mathbf{M}_f &= \frac{1}{N}\sum_{i=1}^{N} f(x,b_i), \\ S_f^2 &= \frac{1}{N-1}\sum_{i=1}^{N}(f(x,b_i)-\mathbf{M}_f)^2. \end{aligned} \tag{25}$$

This approach does not suffer from *point-optimization effect* since the sample $(b_i)_{i=1,\dots,N}$ is generated randomly according to the PDF ρ_B. However, it is well known that this stochastic approach requires a large sample to provide an accurate estimation of the statistics. For CFD applications, a direct Monte-Carlo method is not conceivable presently. Nevertheless, a cheaper approximation $\tilde{f}$ of the cost function can be used in a Monte-Carlo approach to estimate the mean and variance, as suggested in [31]:

$$\begin{aligned} \tilde{\mathbf{M}}_f &= \frac{1}{N}\sum_{i=1}^{N} \tilde{f}(x,b_i), \\ \tilde{S}_f^2 &= \frac{1}{N-1}\sum_{i=1}^{N}(\tilde{f}(x,b_i)-\tilde{\mathbf{M}}_f)^2. \end{aligned} \tag{26}$$

4 GENETIC ALGORITHMS

In *The Origin of Species*[32], Charles Darwin stated the theory of natural evolution. Over many generations, biological organisms evolve according to the principles of natural selection like *survival of the fittest* to reach some remarkable forms of accomplishment. In nature, individuals in a population have to compete with each other for vital resources. Because of such selection, poorly performing individuals have less chance to survive, and the most adapted, or *fit* individuals produce a relatively larger number of offsprings. After a few generations, species evolve spontaneously to become more and more adapted to their environment. In 1975, Holland developed this idea in Adaptation in natural and artificial system. By describing how to apply the principles of natural evolution to optimization problems, he laid down the first Genetic Algorithm. Holland's theory has been further developed and now Genetic Algorithms (GAs) emerge as a powerful adaptive method to solve search and optimization problems. Some results can be seen in [33][34][12].

In Genetic Algorithms, we use the term individual to denote one configuration of the optimal shape. The feasibility of the shape is judged by a fitness function, reflecting the minimized cost functional and penalizing geometrically unfeasible individuals. The shape parameters of one individual are encoded in the individual's chromosome. The Genetic Algorithm operates simultaneously on an entire population of individuals (shapes), the initial population being generated either randomly or as a set of feasible individuals using an a-priori engineering guess. The core of the Genetic Algorithm consists in selection, crossover and mutation operators, whose role is to mimic natural empiric laws of survival of the fittest (selection), their procreations (crossover) and occasional mutations. The generation operators are usually not deterministic, they implement their operators in the probabilistic sense, with a given probability distribution. The crossover is performed with a crossover probability p_{cross} (or crossover rate); two selected individuals (parents) exchange parts of their chromosome to create two offsprings. This operator tends to enable the evolutionary process to move towards *promising* regions of the search space. The mutation operator is introduced to prevent premature convergence to local optima by randomly sampling new points in the search space. It is carried out by flipping bits at random, with some (small) probability p_{mut}.

Genetic algorithms are stochastic iterative processes that are not guaranteed to converge. They have a great potential to explore the whole search space and identify multiple local maxima and to converge to the global optimum while a *point to point* method will generally stall in a local optimum only. They are found more robust in the case of non differentiable, multi-modal or non convex functions, and are particularly interesting for search of a trade-off optimum with respect to several criteria (Pareto front, Nash game).

4.1 Coding

To apply Genetic Algorithms to a specific problem, we must first define an appropriate representation for the solution. We must find a way to encode any solution of the problem into a chromosome of a certain structure. This structure shared by all the chromosomes is called the genetic representation. Solutions might be encoded into a string of bits of a given length. The advantage of bit-string chromosomes is their versatility and simplicity. Historically, GAs have always tried to be a universal solver, able to deal with a wider range of problems. So, binary coding was seen as a standard representation that can fit almost all kinds of search space. Indeed, a string of bits can encode integers, real numbers, sets or whatever is appropriate for the optimized problem. Moreover, the genetic operators over the binary chromosomes are quite simple. Mutation and recombination of bit strings can be done with very simple and universal operators. However, bit strings are often not really appropriate for particular problems. A problem-specific representation, where integer genes represent integer parameters, real genes represent real parameters, character strings represent sets, and so on, can be customized in a way that gives more sense and coherence to the algorithm.

4.2 Fitness Function

Once the genetic representation has been defined, the next step is to associate to each solution (chromosome) a value corresponding to the fitness function. There is generally no problem in determining the fitness function. Particular attention should be taken due to the fact the selection is done according to the fitness of individuals.

In the case of multi-criterion optimization, the fitness function is definitely more difficult to determine. There is often a dilemma as how to determine if one solution is better than another. The trouble comes more from the definition of a *better* solution rather than from how to implement a GA to resolve it. For more advanced problems, it may be useful to consider something like Pareto optimality or other ideas from multi-criterion optimization theory.

The constrains in Genetic Algorithms are introduced mostly by penalization and usually the feasibility region is rapidly detected by Genetic Algorithm process. The penalty method allows constraints to be violated, depending on the magnitude of the violation. However, a penalty that is proportional to the degree of infeasibility is incurred which degrades the objective function. If the penalty is large enough, highly infeasible individuals will rarely be selected for reproduction, and the GA will concentrate on feasible or nearly-feasible solutions.

4.3 Selection

The selection operator is designed to implement the law of the survival of the fittest. Like everywhere in the genetic algorithm, two aspect are to be considered: the convergence of the genetic algorithm towards some solution and the variety (diversity) of the genetic material influencing the exploratory potential for new solutions. Usually, these two aspects act in contradiction. Following this remark, the design of a good selection operator must introduce an algorithm which would handicap bad individuals but would not spoil the variety of the genetic material by choosing systematically only the best individuals to be parents for a new generation.

Typically we can distinguish two types of selection schemes, proportionate selection and ordinal-based selection. Proportionate-based selection picks out individuals based upon their fitness values relative to the fitness of the other individuals in the population. Ordinal based selection schemes select individuals not upon their raw fitness, but upon their relative ordering (ranking) within the population.

4.3.1 Roulette-wheel

Roulette-wheel selection is the classic and popular fitness-proportionate selection. It simply assigns to each solution a sector of a roulette wheel whose size (the angle it subtends) is proportional to the appropriate fitness measure (usually a scaled fitness of some sort). Then it chooses a random position on the wheel (and the solution to which that position was assigned, as if spinning the wheel). In order to create a new population of parents, the weighted roulette-wheel is spun n times, where n is the population size.

4.3.2 Tournament

Tournament selection is an ordinal-based scheme, robust and relatively simple. Many variations exist, but the basic mechanism is to pick k members of the population at random and then select one of them in some very simple manner that depends on fitness. Choosing the best members of the tournament produces a relatively strong selection pressure. So, generally the best is chosen with the probability p, if it fails to be chosen, the second best is chosen with the probability p, and so on until the final solution is chosen. The selection pressure can be adjusted by changing k and p. Clearly tournament selection as described is unaffected by the absolute fitness values, and in effect depends only on the rank of any solution in the population.

4.4 Crossover

It is the process of taking two parent solutions and producing from them a child. There are many different reproducing operators. The most common is an N-point crossover. The N-point crossover takes two chromosomes, aligns them and divides them by N cuts into $N+1$ segments. A child is produced by choosing alternatively a segment from one or the other parent. Two different children can be produced depending if the first segment is taken from the mother or from the father.

Uniform crossover is quite different from the N-point crossover. Each gene (bit) in the offspring is created by copying the corresponding gene from one or the other parent, chosen according to a randomly generated binary crossover mask of the same length as the chromosomes. Where there is a 1 in the crossover mask, the gene is copied from the first parent, and where there is a 0 in the mask the gene is copied from the second parent. A new crossover mask is randomly generated for each pair of parents. Offsprings therefore contain a mixture of genes from each parent.

4.5 Mutation

Mutation is a simple operator consisting of random changes in the value of genes in a chromosome. Mutation has traditionally been considered as a simple search operator. If crossover is supposed to exploit the current solutions to find better ones, mutation is supposed to help for the exploration of the whole search space. Mutation is often seen as a background operator to maintain genetic diversity in the population. There are many different forms of mutation for the different kinds of chromosome coding schemes. For binary coding, a simple mutation can consist in inverting the value of each gene with a small probability.

4.6 Parameters of Configuration

Let us summarize what kind of information one should input to Genetic Algorithms:
- coding: range of variation [min,max] and coding-precision ε, for real coding ε is not necessary,
- choice of genetic operators *selection, crossover, mutation*
- probabilities for stochastic genetic operators
- population size and stopping condition

popsize: the size of the population
maxgen: maximum number of generations
pcross: percentage of crossover
pmutat: percentage of mutation
selecttype: type of selection (tournament or roulette-wheel)
crosstype: type of crossover (N-point, uniform,etc.)

4.7 Multi-Objective GAs[35]

For multi-objective optimization problems, it is necessary to make some modifications to the basic GAs. There exist several variants of GAs for this kind of problem; see for example Vector Evaluated GAs (VEGAs)[36] and Non-dominated Sorting GAs (NSGAs)[37]. For further information on GAs for multi-objective optimization, see Reference [38] and references therein. In the following, we describe the basic ideas of NSGA.

The fitness values are computed using the following procedure:

ALGORITHM: Non-dominated Sorting

```
    Choose a large dummy fitness value $F$;
repeat
    Find the non-dominated individuals among the individuals
      whose fitness values are not set;
    Set the fitness value of individuals found in previous step to $F$;
    Decrease the dummy fitness value;
until (fitness values of all individuals are set)
```

The GAs which we have developed and used is based on NSGA. Since our key idea is to employ the *tournament selection*, it is necessary to make some modifications. The fitness values are computed exactly in the same way as in NSGA. For each tournament, a fixed number of individuals are selected randomly. The individual which has the highest fitness value wins the tournament, i.e., it is selected to be a parent in the breeding. If there are several such individuals then the first one to enter the tournament wins.

Unfortunately, if there were no modifications to the previous tournament selection, the population would usually converge towards one point on the set of Pareto optimal solutions whereas the aim was to obtain several points from the Pareto set. Therefore, some kind of mechanism is required in order to maintain diversity in population. The most obvious way would be to use the fitness value sharing. It has been shown that this approach fails to preserve the diversity in population [39]. Therefore, a modified algorithm is proposed.

Instead of using some previously considered method, we employ a new way to preserve the diversity of the population. We shall call this approach a tournament slot sharing. A sharing function is defined by

$$Sh(d_{ij}) = \begin{cases} 1 - \left(\dfrac{d_{ij}}{\sigma_{share}}\right)^2, if \;\; d_{ij} < \sigma_{share} \\ 0, \qquad\qquad otherwise, \end{cases} \tag{27}$$

where d_{ij} is the genotypic distance between the individuals i and j, i.e., in our case the euclidean distance between the vectors defining the designs i and j. The parameter σ_{share} is the maximum sharing distance for a tournament slot. This very same sharing function $Sh(d_{ij})$ is also used in the classical fitness value sharing. Now the probability for the individual i to enter a tournament is computed using the formula

$$p_i = \frac{1/\sum_{j=1}^{n} Sh(d_{ij})}{\sum_{k=1}^{n}(1/\sum_{j=1}^{n} Sh(d_{ij}))} \tag{28}$$

where the parameter n is the size of population. Hence, this is the same as the roulette wheel selection for the rivals in a tournament. Each individual's slice of roulette wheel is proportional to the inverse of the sum of all sharing functions associated to this individual.

An elitist mechanism is added to our algorithm since it guarantees the cost function values to decrease monotonically from one generation to the next. Also, it usually accelerates the convergence. This is implemented by copying from the old population to the new population all the individuals which would be non-dominated in the new population. Hence, the number of copied individuals varies from a generation to another. As a coding, we have used the floating point coding [40]. This is a rather natural choice, since the design is defined by a vector of floating point numbers. The crossover is made using one crossover site.

The mutation utilizes a special distribution promoting small mutations. More precisely, the mutation is performed in the following way for a single string: The floating point numbers of the string under consideration are gone through one by one. Let us assume that x is one of these numbers and it is to be mutated. Let l and u be the lower and upper limits for x. The mutated x is denoted by x_m and it is computed as follows:

1. Set $t = (x-l)/(u-l)$;
2. Compute

$$t_m = \begin{cases} t - t\left(\dfrac{t - rnd}{t}\right)^p, & rnd < t, \\ t, & rnd = t, \\ t + (1-t)\left(\dfrac{rnd - t}{1-t}\right)^p, & rnd > t \end{cases} \tag{29}$$

where rnd is a random number from the closed interval $[0, 1]$;

3. Set $x_m = (1 - t_m)l + t_m u$.

In step 2, the parameter p defines the distribution of the mutation. We call this parameter the mutation exponent. If $p = 1$ then the mutation is uniform. The probability of small mutations grows as the value of p grows. Hence, we are ready to present the following GA and the parent selection procedure:

ALGORITHM: The Modified NSGA

```
Initialize population:
Compute object functions (in parallel);
$do\ i = 2, number-of-generations$
        Compute fitness values using non-dominated sorting;
        Compute probabilities for each individual to enter tournament;
        $repeat$
                Select two parents;
                Form two childrens using crossover;
        $until$ (new population is full);
        Perform mutation;
        Compute object functions (in parallel);
        Copy individuals from old population according to elitism;
```

```
  enddo
ALGORITHM: Select Parent
  repeat
      Select one individual to tournament using the probabilities $p_i$ in (28);
  until (tournament is full );
  Find best the individual from the tournament according to the fitness values.
```

4.8 Conclusion

Since about fifteen years ago, Genetic Algorithms have been introduced in aerodynamics shape design problems (see [41] [42][43]. The main concern related to the use of genetic algorithms is the computational effort needed for the accurate evaluation of a configuration that might lead to unacceptable computer time if compared with more classical algorithms. Eventhough, fitness function value can be effectively estimated approximately by using Response Surface Modelling.

5 2D NS SOLVER ON UNSTRUCTURED MESH

We solve the 2D Navier-Stokes equations by using a Finite-Volume Galerkin method on unstructured meshes. 2D unstructured mesh is generated by pre/post-processing software **GID** of CIMNE. To solve the Euler part of the equations, a Roe scheme is used. To compute turbulent flows a $k - \varepsilon$ model is chosen. Near-wall turbulence is computed by a two-layer approach. Time dependant problems can also be considered as a fourth order Runge-Kutta solver has been used.

Following figure 1 shows the 2D unstructured mesh around a three element airfoil generated by using ***GID***. Figure 2 shows the iso-pressure plotting of the flow over this three element airfoil at $M_\infty = 0.25, \alpha = 15^0$ and Reynolds number $Re = 1.8 \times 10^6$. In order to valid the solver further, we analyze the viscous flow over the $RAE2822$ airfoil in subsonic flow, very fine mesh within boundary layer is generated, see Figure 3; the corresponding velocity profile within the boundary layer is shown in Figure 4, fly condition is $M_\infty = 0.35, \alpha = 2^0$ and $re = 2.0 \times 10^6$; and Figure 5 is the iso-mach plotting, which denotes apparently that low speed flow region is exist with boundary layer due to the viscosity.

6 RESPONSE SURFACE MODELLING METHODS[44]

6.1 Numerical Noise Issues

As noted, the existence of high frequency low amplitude numerical noise pervades many computational efforts. Numerical noise poses a significant hindrance to gradient-based optimization because it inhibits the use of finite difference methods in gradient estimation. Typical sources of numerical noise include

1. the incomplete convergence of iterative processes,
2. the use of adaptive numerical algorithms,
3. round-off errors, and
4. the discrete representation of continuous physical objects.

When confronted with numerical noise in optimization problems, one may elect to elimi-

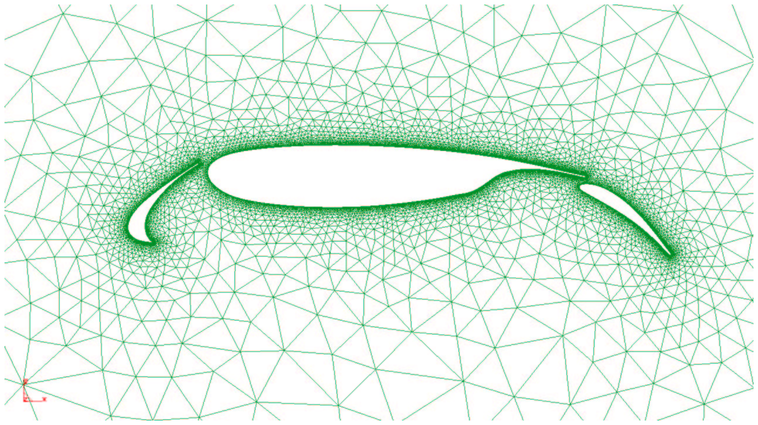


Figure 1: 2D unstructured mesh around three element airfoil

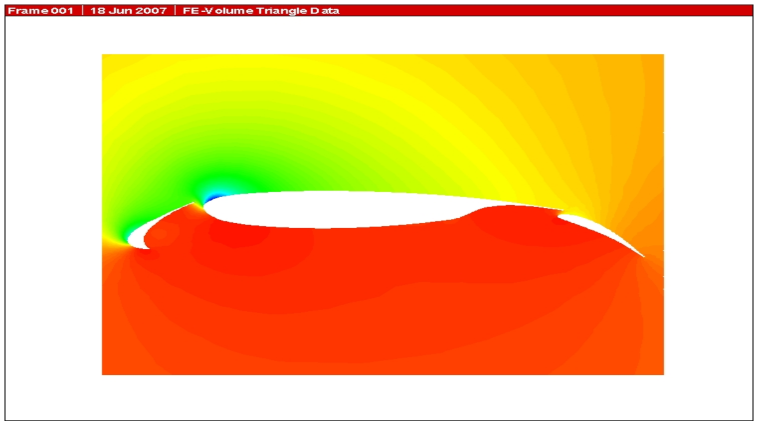


Figure 2: Iso-pressure cloud plotting around three element airfoil in subsonic flow

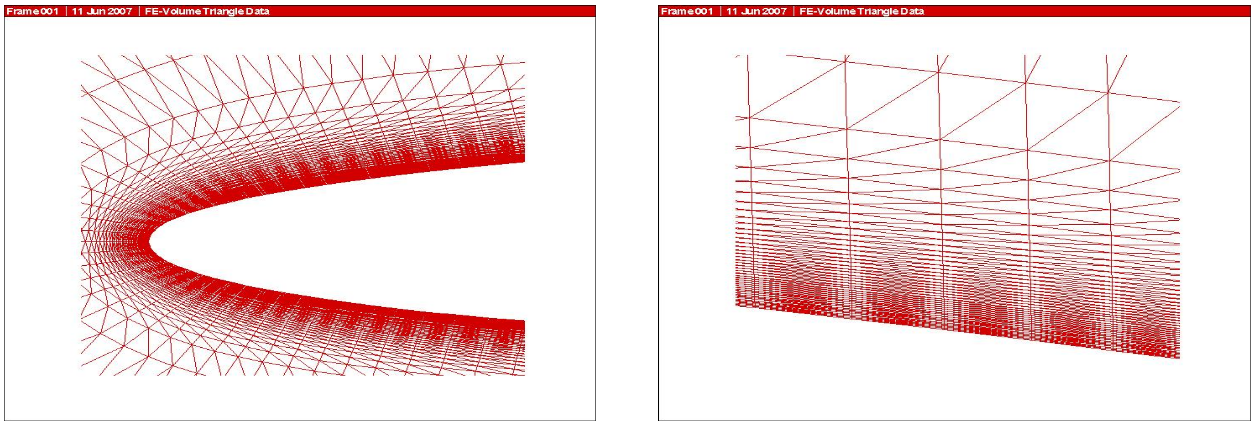
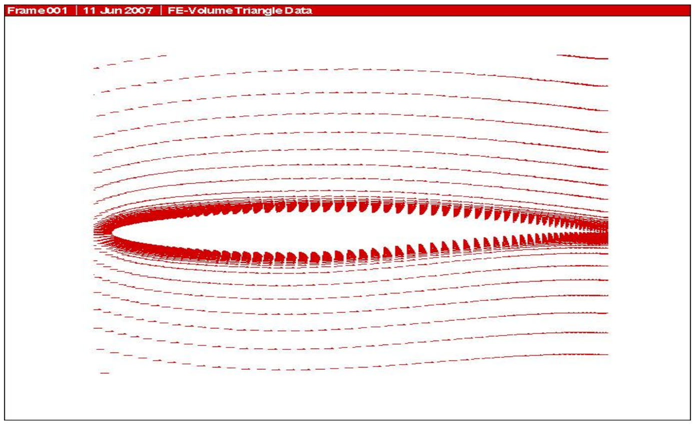
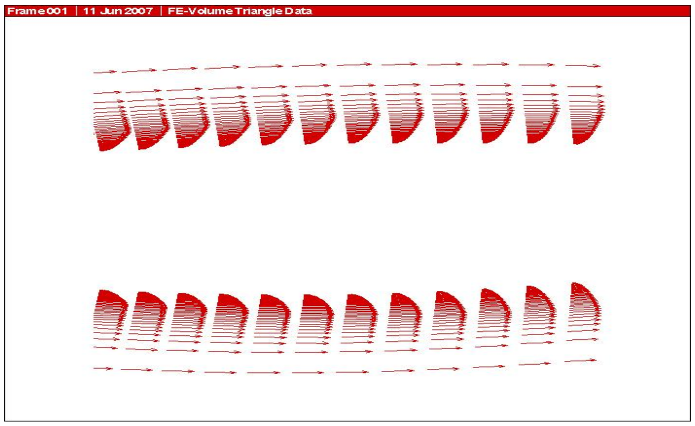

Frame 001 | 11 Jun 2007 | FE-Volume Triangle Data
Frame 001 | 11 Jun 2007 | FE-Volume Triangle Data

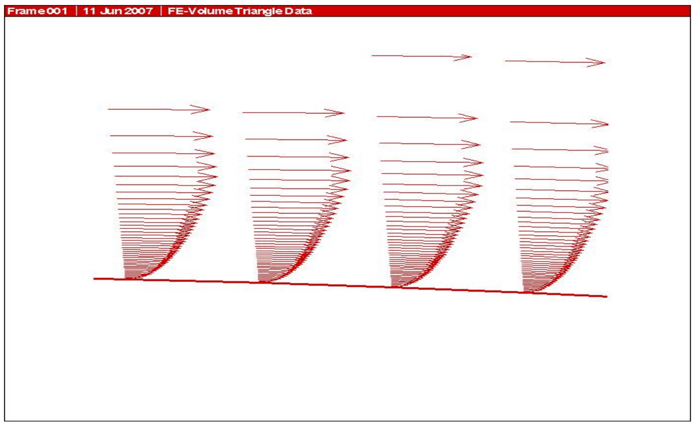
Figure 3: Partial view of boundary layer mesh around $RAE2822$ airfoil



a. velocity profile of the flow



b. partial view of velocity profile



c. zoom in the velocity profile

Figure 4: Velocity profile within the boundary layer

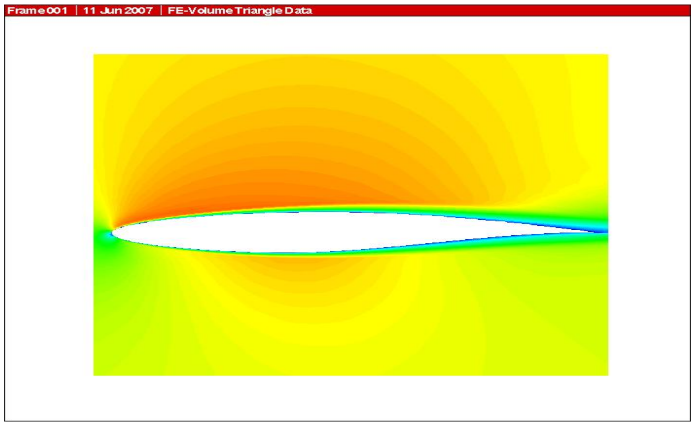


Figure 5: Iso-Mach cloud plotting around $RAE2822$ airfoil in subsonic flow

nate the noise sources (e.g., problem reformulation or software modification) or one may use optimization methods that are insensitive to numerical noise. While eliminating the noise sources is an attractive option, it is not always possible to reformulate the optimization to accomplish this goal. Eliminating noise sources through software modification is also possible, but this can be an extremely tedious task, particularly so if the user is not the original software developer. Furthermore, in many industrial settings the user does not have access to the source files for commercially available software. The development of optimization methods which are insensitive to numerical noise is an active area of research in the optimization community. While the efforts of Gilmore and Kelley [45] are promising, their methods are not applicable when the evaluation of the objective function and/or the evaluation of the constraints are computationally expensive. The Variable Complex Response Surface Modelling (VCRSM) method is a hybrid of the two approaches used to eliminate numerical noise in optimization problems. See section 5.3 and 5.4 below for more details on the development and use of VCRSM techniques.

6.2 Description of the VCRSM Method

The VCRSM method removes sources of numerical noise which inhibit numerical optimization. This is accomplished by creating smooth, polynomial response surface models or Bayesian interpolating models which replace the original analysis methods that produce numerical noise. Without numerical noise to inhibit optimization, there is a greater probability that the globally optimal design will be found, regardless of the initial variable values used at the start of the numerical optimization.

One of the most important advantages obtained by using response surface models in optimization is significant reduction in computational expense. This allows the user to perform global optimization and reliability-based optimization, which are otherwise prohibitively computationally expensive. In addition, the use of response surface models allows the design engineer to quickly perform a variety of trade-off studies which provide information on the sensitivity of the optimal aircraft design to changes in performance criteria and to off-design conditions.

The reduction in the computational expense of optimization when using response surface models motivates their use in the modelling of data obtained from expensive analysis methods, even though the expensive analysis methods may not produce numerical noise.

6.3 RSM and Physical Experiments

Response surface modelling methods originally were developed to analyze experimental data and to create empirical models of the observed response values. The particular forte of RSM is its applicability to investigations where there are few observations because the physical experiment is both very expensive and very time consuming to perform.

In RSM the relationship between observations and independent variables is defined as

$$y = f(\mathbf{x}) \tag{30}$$

where y is the observed response, $\mathbf{x}$ is the vector of n_v independent variables defined as

$$\mathbf{x} = [x_1, x_2, ..., x_{n_v}] \tag{31}$$

and $f(\mathbf{x})$ is the unknown function. The empirical model of the unknown function found via the application of RSM is defined as

$$\hat{y} = \hat{f}(\mathbf{x}) \tag{32}$$

where $\hat{f}(\mathbf{x})$ typically is a first or second order polynomial in $\mathbf{x}$. Note that the random error (uncertainty) present in stochastic experimental data is implicit in both $f(\mathbf{x})$ and $\hat{f}(\mathbf{x})$.

RSM employs the statistical techniques of regression analysis and Analysis Of Variance (ANOVA) to determine $\hat{f}(\mathbf{x})$ through a systematic decomposition of the variability in the observed response values. The empirical model is then estimated by assigning portions of the variability to either the effect of an independent variable or to random error.

6.4 Polynomial Models for RSM

In many RSM applications, either linear or quadratic polynomials are assumed to accurately model the observed response values. Although this is certainly not true for all cases, RSM becomes prohibitively expensive when cubic and higher-order polynomials are chosen for experiments involving several variables. In addition, cubic and higher-order polynomial models may contain one or more inflection points. In gradient-based numerical optimization schemes the optimizer may converge to an inflection point rather than to a local or global optimum.

If n_s analysis are conducted and $p = 1, ..., n_s$, then a quadratic response surface (RS) model has the form

$$y^{(p)} = c_0 + \sum_{1 \le j \le n_v} c_j x_j^{(p)} + \sum_{1 \le j \le k \le n_v} c_{(n_v - 1 + j + k)} x_j^{(p)} x_k^{(p)}, \tag{33}$$

where $y^{(p)}$ is the response; $x_j^{(p)}$ and $x_k^{(p)}$ are the n_v design variables; and c_0, c_j, and $c_(n_v - 1 + j + k)$ are the unknown polynomial coefficients. Note that there are $n_t = (n_v + 1)(n_v + 2)/2$

coefficients (i.e., model terms) in the quadratic polynomial. This polynomial model may be written in matrix notation as

$$y^{(p)} = C^T \bar{X}^{(p)}, \tag{34}$$

where C is the vector of length n_t of unknown coefficients to be estimated,

$$C = [c_0, c_1, ..., c_{n_t-1}], \tag{35}$$

and $\bar{X}(p)$ is the vector of length n_t corresponding to the form of the $x_j^{(p)}$ and $x_k^{(p)}$ terms in the polynomial model (33). For the p^{th} observation this is

$$\bar{X}^{(p)} = [1, x_1^{(p)}, x_2^{(p)}, ...x_{n_v}^{(p)}, (x_1^{(p)})^2, x_1^{(p)}x_2^{(p)}, ..., (x_{n_v}^{(p)})^2]. \tag{36}$$

Note that there is a difference between the p^{th} vector of independent variables, $X^{(p)}$, and the p^{th} vector of independent variables mapped into the form of the polynomial model, $\bar{X}^{(p)}$.

Estimating the unknown coefficients requires n_s analysis, where $n_s \geq n_t$. Under such conditions, the estimation problem may be formulated in matrix notation as

$$Y \approx XC, \tag{37}$$

where Y is the vector of n_s observed response values,

$$Y = [y^{(1)}, y^{(2)}, ..., y^{(n_s)}], \tag{38}$$

and X is the matrix formed by the n_s row vectors $\bar{X}^{(p)}$ which is assumed to have rank n_t. Thus, X may be expressed as

$$X = \begin{bmatrix} 1 & x_1^{(1)} & x_2^{(1)} & \dots & (x_{n_v}^{(1)})^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_1^{(n_s)} & x_2^{(n_s)} & \dots & (x_{n_v}^{(n_s)})^2 \end{bmatrix}. \tag{39}$$

The unique least squares solution to Equation (37) is

$$\hat{C} = \left(X^T X\right)^{-1} X^T Y, \tag{40}$$

where $\left(X^T X\right)^{-1}$ exists if the rows of X are linearly independent. When $\hat{C}$ is substituted for C into Equation (34), values of the response may be predicted at any location $\mathbf{x}$ by mapping $\mathbf{x}$ into $\bar{X}^{(p)}$. In matrix notation this corresponds to

$$\hat{Y} = \hat{C}^T \bar{X}^{(p)}. \tag{41}$$

Note that if $n_s > n_t$ the system of equations is overdetermined. Thus, the predicted response values (from the polynomial model) at the original sample locations may differ from the observed response values at the sampled locations.

Polynomial RS models can be thought of as *global* models in which all of the n_s observed values of the response are equally weighted in the fitting of the polynomial surface. At an unsampled location in design space, $\mathbf{x}$, response observations that are near to $\mathbf{x}$ (in the sense of Euclidean distance) have an equal influence on the predicted response, $\hat{f}(\mathbf{x})$, as do the response observations that are far from $\mathbf{x}$. It can be argued that such a global model may not be the best approximator if the true unknown response has many real local optima (as opposed to the artificial local optima created by numerical noise). In such a situation an approximation scheme having *local* modelling properties may be more attractive, i.e., where $\hat{f}(\mathbf{x})$ is more strongly influenced by nearby measured response values and is less strongly influenced by those further away. Such local modelling behavior is characteristic of interpolation models, of which DACE models are one particular implementation.

6.5 Approximation Test Problems and Results

6.5.1 Test problem formulation

A simple test function was chosen so that it could be exhaustively examined with minimal computational expense. This test function has the form

$$y(\mathbf{x}) = \sum_{i=1}^{n_v} \left[\frac{3}{10} + \sin(\frac{16}{15}x_i - \epsilon) + \sin^2(\frac{16}{15}x_i - \epsilon) \right], \tag{42}$$

where the term ϵ acts as a shifting mechanism to make the response, $y(\mathbf{x})$, appear more or less quadratic on the range $[-1, 1]^{n_v}$. The values used for ϵ are described below.

The first test function (case 1) was created using $\epsilon = 0.$, this function has a quasi-quadratic trend on $[-1, 1]$.

The second test function (case 2) was created for $\epsilon = 1.0$, this function appears a quasi-sinusoidal on $[-1, 1]$.

6.5.2 Approximation results

Result of case1: For this case, the function (42) has a quasi-quadratic trend on $[-1, 1]$. The analytical function is shown in Figure 6 for $n_v = 2$, and approximated surface by RSM is shown in Figure 7, we can see that both approximated and original surfaces are well agreement because the quadratic like the function is, see Figure 8.

Result of case2: For this case, the function (42) appears quasi-sinusoidal on $[-1, 1]$ shown in Figure 9, but the RSM approximation is quadratic surface shown in Figure 10. So, there is quite difference between the estimated and original surface, see Figure 11.

In order to increase the approximation accuracy of RSM for estimating any kind of function, we split the design space into a set of smaller subspaces, then to approximate response value in each subspace by RSM methods and enforce the continuity on the interfaces between the subspace, see Figure 12, 13 and 14.

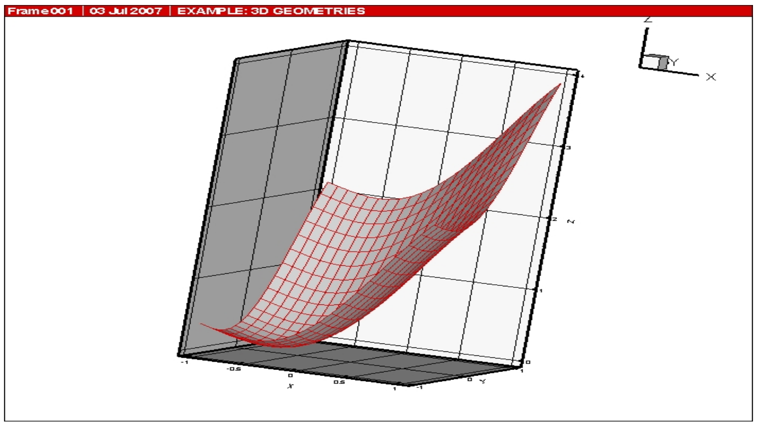


Figure 6: Original surface of test case 1

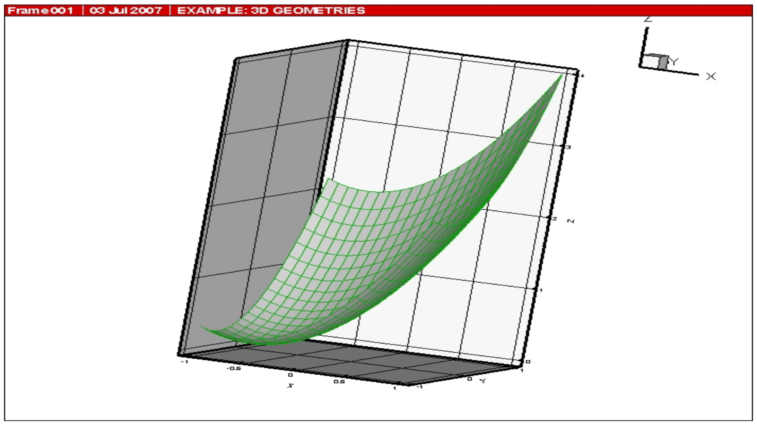


Figure 7: Estimated surface by RSM for test case 1

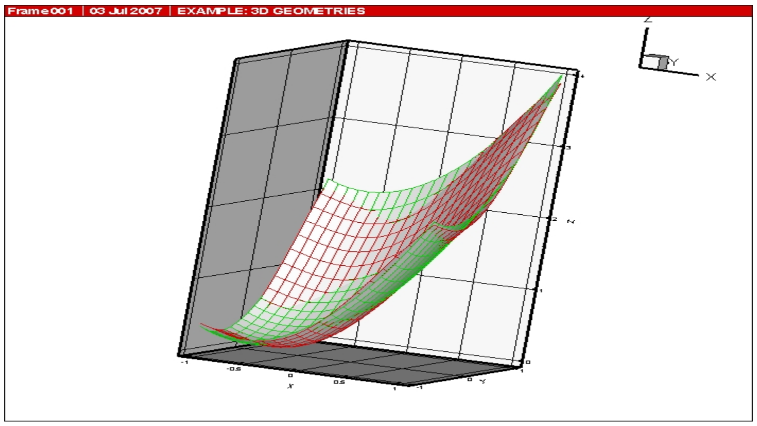


Figure 8: Comparison of estimated surface with original one for test case1

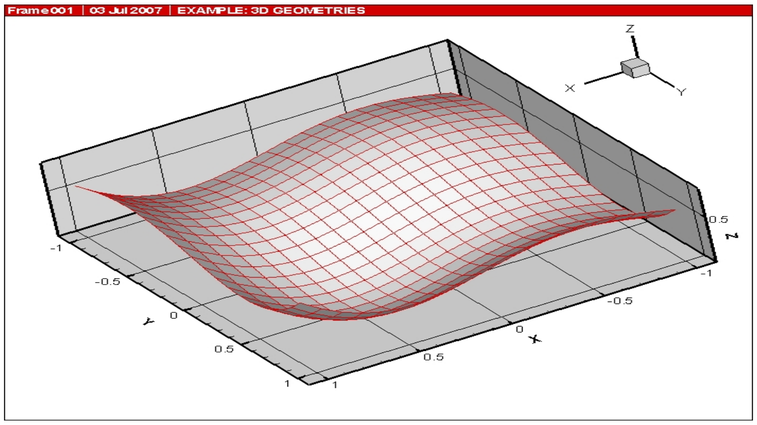


Figure 9: Original surface of test case 2

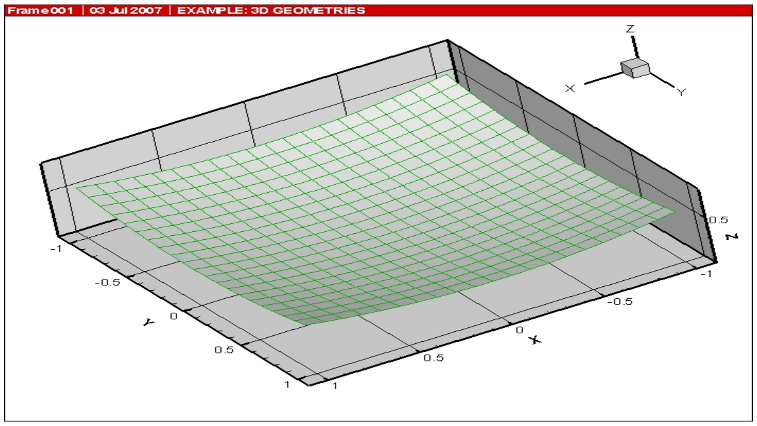


Figure 10: Estimated surface by global RSM approximation for test case 2

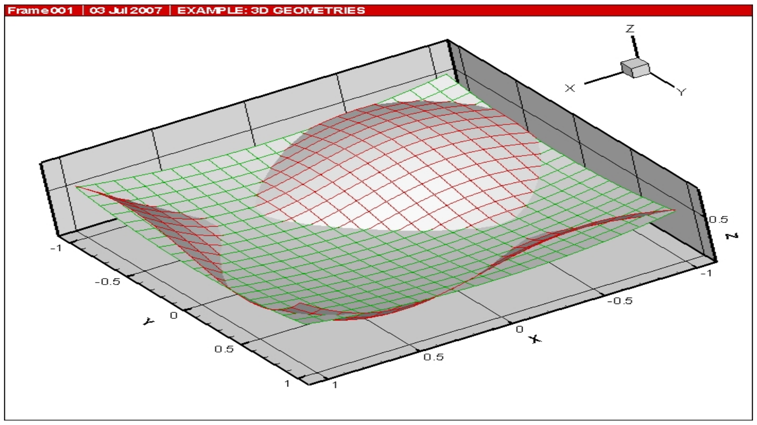


Figure 11: Comparison of estimated surface with original one for test case2

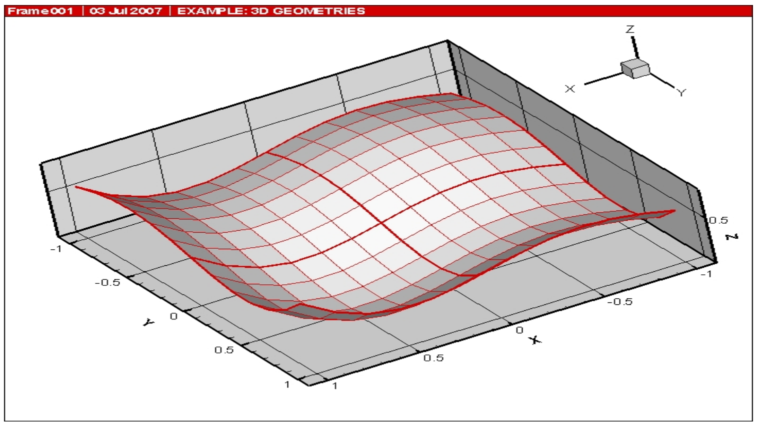


Figure 12: Split the original surface of test case 2 into four pieces

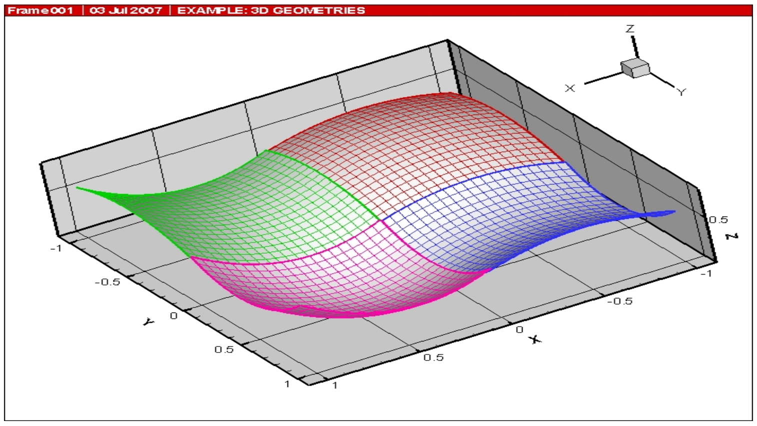


Figure 13: Approximate each pieces by RSM method for test case 2

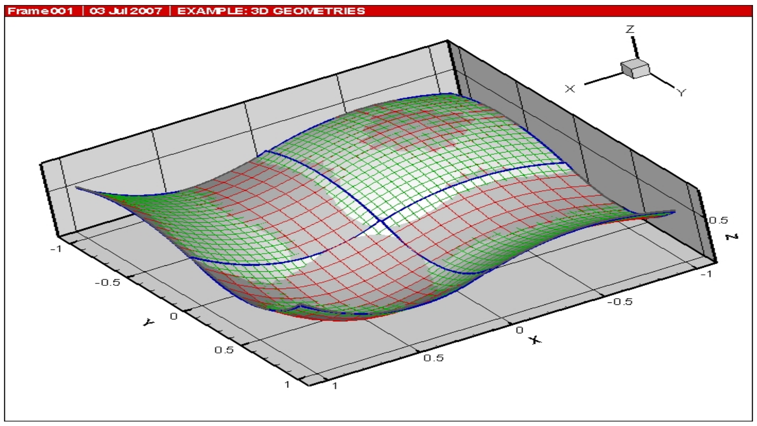


Figure 14: Comparison of estimated surface with original one for test case2 by locally approximation

7 SEMI-TORSINAL SPRING ANALOGY FOR MESH MOVEMENT[46]

7.1 Introduction

Flow problems with moving boundaries arise in many engineering areas, such as Fluid-Structure Interaction (FSI), free surface flows and bio-fluid mechanics. These flows often occur in 2D/3D spatial domains characterized by complex configurations and non-uniform deformations. In all applications, a CFD analysis must either track interfaces (e.g. by the VOF technique [47]) or must cause the computational mesh to conform to the changing flow domain. This report is concerned with the second approach, which is most common for flows without free surfaces.

A priori, the simplest approach is to generate the mesh automatically at each time step on the time dependent domain. Unfortunately, in dynamic 2D/3D domains with realistic configurations, neither structured nor unstructured mesh generation is a practical approach, mainly due to high CPU cost, and the difficulties in automation and mesh quality control [48][49][50]. Cartesian grid generation is simple and can be automatic, but mesh quality near the surface is poor [48][49]. In addition to the difficulties associated with mesh generation, extra-computational cost is entailed because of mapping the solution from the old mesh to the new mesh, which, particularly for unstructured meshes, is not trivial.

To overcome these difficulties, algorithms that deform an existing mesh to conform to the dynamic geometry have been proposed, such as transfinite interpolation (TFI), isoparametric mapping, the elastic analogy, and the spring analogy [51][52][53][54]. TFI and isoparametric mapping are algebraic techniques suitable for structured meshes in simple domains, while elastic analogy and spring analogy methods are applicable to both structured and unstructured meshes. The elastic analogy treats a deforming spatial domain as a mass of elastic material so that boundary displacements are spread into the mesh through elastic forces. The discrete version of the elastic analogy is the spring analogy, in which an existing mesh is viewed as a network of fictitious springs and boundary displacements propagate into the mesh by virtue of a static equilibrium condition requiring that the sum of spring forces at each internal node be zero, see Figure 15 left panel. These algorithms are often referred to as moving/dynamic mesh algorithms. A dynamic mesh conforms to the changing configuration of a computational domain, and the number of nodes and the mesh's connectivity remain the same through the course of deformation. When such a dynamic mesh algorithm is implemented in conjunction with the Navier-Stokes (NS) equations in an arbitrary Lagrangian-Eulerian (ALE) formulation, solution mapping from old mesh to new mesh is avoided [55]. Among the moving mesh algorithms, the spring analogy is widely used because of its simplicity, modest memory requirements, and flexibility in local adjustment of spring stiffness [56][57][58][59][60][61][62].

Batina [51] proposed a spring analogy model to update a 2D unstructured triangular finite element mesh in a study of Euler flow around an oscillating airfoil. The spring stiffness for a given edge was taken to be inversely proportional to the length of the edge and the spring force was proportional to the difference of the displacements at the two nodes connected by the spring. This spring analogy model has been widely adopted for mesh updating in solving moving boundary problems. It is referred to as a lineal spring analogy model because only

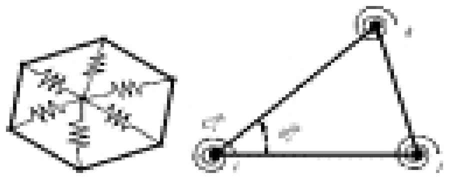


Figure 15: Left panel, spring analogy: each edge of the mesh is modelled as a fictitious spring, and boundary deformation is spread into the mesh through the spring forces. Right panel, torsional spring analogy: each vertex is attached a torsional spring that resists the change of the angle at the vertex. The moments generated by these torsional springs are proportional to the torsional stiffness and the angular displacement

edge length is taken into consideration in the formulation of spring stiffness.

A major drawback of this approach is that element inversion may occur [52][54], because there is no constraint on the angles between two adjacent edges in an element. For most computational algorithms, element inversion must be avoided at all costs. A torsional spring analogy model for unstructured meshes has been therefore developed in 2D [54] and extended to 3D [53]. In its 2D version, additional torsional springs are attached to each vertex of the triangular elements; these torsional springs produce torques that resist angular displacements on an element. Mesh regularity is preserved even when the deformation is large.

Unfortunately, this approach requires a kinematic formulation based on the assumption of sufficiently small displacements and rotations, as well as a force transformation that converts the moments generated by torsional springs to spring forces at each vertex so as to incorporate the torsional springs into existing lineal spring analogy models. Extending the model to 3D leads to a rather complex formulation and appreciable computational overhead [53].

An improvement of the above approach is the semi-torsional spring analogy model, which has been developed for 2D meshes [52]. In this approach, the lineal stiffness of an edge on a triangle is divided by the opposite angle in the triangular element, to obtain a *semi-torsional stiffness*. Neither a displacement formulation nor a force transformation is necessary in this semi-torsional model. However, the model only handles 2D triangular meshes and fails for 3D problems. For example, the semi-torsional stiffness does not sense element inversion if it is derived by dividing the lineal stiffness by the angles formed between the edges of any given face of a tetrahedron, because these angles can remain well-behaved even as the element inverts by a vertex passing through the face opposing it.

In view of the above, there is a need for a simple, robust and computationally efficient scheme for maintaining element quality during mesh deformation with the spring analogy approach. This scheme must work in both 2D and 3D, be able to handle large deformations, and work well for fully unstructured meshes. Paper [46] presented such a scheme, We use it in our optimization to perform 2D Multi-element unstructured mesh movement due to the slat/flap position change.

7.2 Methods

7.2.1 Lineal spring analogy

Spring analogy models can be categorized into two types: vertex models and segment models [52]. Vertex springs are always under tension unless the spring length is zero, and are mainly used for mesh smoothing because the tension tends to pull the mesh towards equal inter-nodal distances. On the other hand, segment springs have zero tension at some equilibrium length (before deformation) and nodal displacements on the boundary create spring forces that subsequently displace internal nodes. Segment models are better suited for moving internal nodes to follow a dynamically deforming computational domain, and only segment spring analogy models are used in this paper.

The lineal spring stiffness k_{ij} for a given element edge $i-j$ takes the following general form:

$$k_{ij}^n = \frac{\lambda}{[(y_{1,i}^n - y_{1,j}^n)^2 + (y_{2,i}^n - y_{2,j}^n)^2 + (y_{3,i}^n - y_{3,j}^n)^2]^\beta}, \tag{43}$$

where the superscript n denotes time step, $(y_{1,i}^n, y_{2,i}^n, y_{3,i}^n)$ and $(y_{1,j}^n, y_{2,j}^n, y_{3,j}^n)$ are the spatial coordinates of the two nodes connected by the edge $i-j$ at time step n, and λ and β are coefficients. The fictitious spring force $\overrightarrow{F}_{ij}^n$ acting on node i from edge $i-j$ is

$$\overrightarrow{F}_{ij}^n = k_{ij}^n(\overrightarrow{\delta}_j^n - \overrightarrow{\delta}_i^n), \tag{44}$$

where $\overrightarrow{\delta}_j^n$ and $\overrightarrow{\delta}_i^n$ are nodal displacements of node j and i at step n, respectively

$$\begin{aligned} \overrightarrow{\delta}_j^n &= \overrightarrow{y}_j^n - \overrightarrow{y}_j^{n-1} \\ \overrightarrow{\delta}_i^n &= \overrightarrow{y}_i^n - \overrightarrow{y}_i^{n-1} \end{aligned} \tag{45}$$

The static equilibrium equation for node i at time step n is

$$\sum_{j=1}^{NE_i} k_{ij}^n \left(\overrightarrow{\delta}_j^n - \overrightarrow{\delta}_i^n\right) = 0, \tag{46}$$

where N_{ei} is the number of nodes directly connected to node i through fictitious springs. A system of equations is derived through applying the equilibrium equation to all nodes in the mesh. To linearize the system, the spring stiffness is computed using nodal coordinates of the previous time step. With displacements on boundary nodes prescribed, and nodal coordinates of the previous time step known, the system is solved iteratively for nodal displacements at internal nodes. Nodal coordinates are updated by using the nodal displacements and old coordinates:

$$\overrightarrow{y}_i^n = \overrightarrow{y}_i^{n-1} + \overrightarrow{\delta}_i^n. \tag{47}$$

The coefficient β is often taken to be 0.5, which means that the stiffness is inversely proportional to the length of the edge. Node collision is prevented by selecting the spring stiffness as the inverse of the edge length, as demonstrated by Blom [52] in a 1D analysis. It is noted that the 2D version of Eq. (43) takes the original form as proposed by Batina [51] when $\lambda = 1$ and $\beta = 0.5$.

7.2.2 Torsional spring analogy

Neither angular displacement nor area (volume) change of an element is associated with its edges' stiffness, which means that element inversion and near-inversion are not sensed by the lineal model. To prevent element inversion on a 2D unstructured triangular mesh, a torsional spring is attached to each vertex in addition to the lineal springs (Fig. 15 right panel) [54]. The stiffness of the torsional spring is defined as

$$C_i^{ijk} = \frac{k}{sin^2\theta_i^{ijk}}, \tag{48}$$

where C_i^{ijk} is the torsional stiffness associated with the angle θ_i^{ijk} that has node i as its vertex on the triangle $\triangle_{ijk}$. The moments generated by these torsional springs are proportional to the torsional stiffness and to angular displacements. By virtue of the definition for torsional stiffness, as θ_i^{ijk} approaches zero or π, the corresponding torsional stiffness increases rapidly and prevents element inversion.

However, displacements and rotations are assumed to be sufficiently small in this 2D model to obtain a linear formulation that expresses the angular displacements in terms of nodal displacements [54]. There are three angles on a 2D element (triangle) and six components of nodal displacements. The linear formulation requires calculation of the corresponding coefficients on each element. The model also requires a force transformation that converts the moments generated by the torsional springs to spring forces on the nodes, so that the equilibrium equations admit torsional force terms in addition to the lineal spring force terms. On a triangular element, six torsional force components need to be associated with three angular displacements. Therefore, successive small deformations are needed to achieve a large deformation.

7.2.3 Semi-torsional spring analogy[46]

A semi-torsional spring analogy model is similar to the lineal formulation, with angle information incorporated into the spring stiffness. Neither displacement formulation nor force transformation is needed, and this approach is therefore easy to implement[46]. For 2D triangular elements, a semi-torsional stiffness of an edge $i-j$ was proposed by Blom [52].

$$k_{ij}^{semi-torsional} = \frac{k_{ij}^{lineal}}{\theta}, \tag{49}$$

where θ is the angle facing the edge on an element. However, this semi-torsional model is not directly applicable to 3D elements. Moreover, an internal edge in a 2D triangular mesh is attached to two elements, and faces two angles which are usually different in magnitude. The above definition gives different stiffness values to a single edge when it is considered on each of its two attached elements.

To deform 2D/3D unstructured meshes for solving moving boundary problems, we propose a semi-torsional spring analogy model based on Zeng's previous work [63], in which the stiffness of an edge is defined as the sum of its lineal stiffness and its semi-torsional stiffness, with the semi-torsional stiffness depending on the angle facing the edge, i.e.

$$\begin{aligned} k_{ij} &= k_{ij}^{lineal} + k_{ij}^{semi-torsional} \\ k_{ij}^{semi-torsional} &= k \sum_{m=1}^{NE_{ij}} \frac{1}{sin^2 \theta_m^{ij}} \end{aligned} \tag{50}$$

where the lineal stiffness is defined as in Eq. (43), NE_{ij} is the number of elements sharing edge $i - j$, and θ_m^{ij} is the facing angle, defined as the angle that faces the edge $i - j$ on the mth element attached to the edge. k is a coefficient having the dimension of the stiffness. In all our numerical experiments, we set the value of the coefficient to be 1.0. On a tetrahedron with vertices i, j, k and l, the angle that faces the edge $i - j$ is taken as the angle formed between triangle $\triangle^{ikl}$ and triangle $\triangle^{jkl}$ (Fig. 16, left panel). The facing angle θ_m^{ij} is easily computed based on the coordinates of the vertices. Let $\overrightarrow{n}_1$ and $\overrightarrow{n}_2$ be the normal vectors of the faces $\triangle^{jkl}$ and $\triangle^{ikl}$, respectively, i.e.

$$\begin{aligned} \overrightarrow{n}_1 &= (\overrightarrow{y}_l - \overrightarrow{y}_j) \times (\overrightarrow{y}_k - \overrightarrow{y}_j) \\ \overrightarrow{n}_2 &= (\overrightarrow{y}_l - \overrightarrow{y}_i) \times (\overrightarrow{y}_k - \overrightarrow{y}_i) \end{aligned} \tag{51}$$

where $\overrightarrow{y}$ denotes the position vector of a vertex. Then

$$sin^2 \theta_m^{ij} = 1 - \frac{(\overrightarrow{n}_1 \cdot \overrightarrow{n}_2)^2}{|\overrightarrow{n}_1|^2 \, |\overrightarrow{n}_2|^2}. \tag{52}$$

By substituting (50) into (44), the spring forces on nodes i and j are expressed as

$$[F_{ij}] = \left(\frac{\lambda}{l_{ij}} + k \sum_{m=1}^{NE_{ij}} \frac{1}{sin^2 \theta_m^{ij}} \right) [B] \, [u_{ij}] \,, \tag{53}$$

where $[F_{ij}] = [F_{ix}, F_{iy}, F_{iz}, F_{jx}, F_{jy}, F_{jz}]^T$, $[u_{ij}] = [u_i, v_i, w_i, u_j, v_j, w_j]^T$ are vectors of spring forces and displacements at nodes i and j , $[B]$ is a 6×6 matrix whose elements are given by $B_{pq} = -\delta_{pq} + \delta_{p,q+3} + \delta_{p+3,q}$, with $\delta_{pq} = 1$ if $p = q$ and $\delta_{pq} = 0$ if $p \neq q$.

For a 2D triangular mesh (Fig. 11, right panel), one edge within the mesh shares two elements and Eq. (10) simplifies to

$$k_{ij}^{semi-torsional} = k \left(\frac{1}{sin^2 \theta_1} + \frac{1}{sin^2 \theta_2} \right), \tag{54}$$

where θ_1 is the angle facing edge $i - j$ on the triangle $\triangle_{ijl}$, θ_2 is the angle facing edge $i - j$

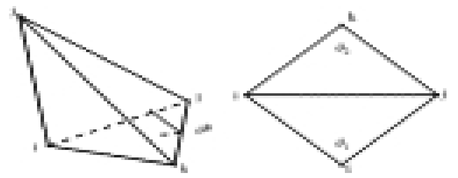


Figure 16: Definition of facing angles in tetrahedral and triangular elements. Left panel: the facing angle θ_{ij}, defined as the angle formed between face jkl and face ikl, faces the edge $i-j$ in a tetrahedral element. In an unstructured mesh, the total number of such angles equals the number of elements attached to edge $i-j$. Right panel: analogous situation in 2D is simpler, since there are always two angles facing one interior edge $i-j$.

on $\triangle_{ijk}$ and k, l, i and j are the vertices of the elements. In this case, we have

$$\begin{aligned} sin^2\theta_1 &= \frac{\left|(\overrightarrow{y}_i - \overrightarrow{y}_l) \times (\overrightarrow{y}_j - \overrightarrow{y}_l)\right|^2}{\left|\overrightarrow{y}_i - \overrightarrow{y}_l\right|^2 \left|\overrightarrow{y}_j - \overrightarrow{y}_l\right|^2}, \\ sin^2\theta_2 &= \frac{\left|(\overrightarrow{y}_i - \overrightarrow{y}_k) \times (\overrightarrow{y}_j - \overrightarrow{y}_k)\right|^2}{\left|\overrightarrow{y}_i - \overrightarrow{y}_k\right|^2 \left|\overrightarrow{y}_j - \overrightarrow{y}_k\right|^2}. \end{aligned} \tag{55}$$

In a 2D triangular mesh, spring forces on the edge $i-j$ are

$$[F_{ij}] = \left(\frac{\lambda}{l_{ij}} + k\left(\frac{1}{sin^2\theta_1} + \frac{1}{sin^2\theta_2}\right)\right)[B^*]\,[u_{ij}]\,, \tag{56}$$

where $[F_{ij}] = [F_{ix}, F_{iy}, F_{jx}, F_{jy}]^T$, $[u_{ij}] = [u_i, v_i, u_j, v_j]^T$, and $[B^*]$ is a 4×4 matrix whose elements are given by $B^*_{pq} = -\delta_{pq} + \delta_{p,q+2} + \delta_{p+2,q}$.

With the above definition for semi-torsional stiffness, an angle approaching 0 or π makes the edge facing this angle very stiff, which prevents further change in the angle and thus avoids element inversion.

In comparison with the semi-torsional model as in Eq. (49), the 2D version of the semi-torsional stiffness as in Eq. (54) has three advantages. First, an edge in a mesh is assigned a definite stiffness value by taking both facing angles in the definition. Second, the semi-torsional stiffness is computationally efficient. It needs only evaluations of the sine of the facing angles, while the magnitude of the facing angle is required in Eq. (49). This magnitude has to be derived through computing both sine and cosine values of the angle from the coordinates of the vertices. Third, extension to 3D using the semi-torsional stiffness is easily achieved by defining a facing angle for an edge in a tetrahedral element.

Compared against the torsional model, the semi-torsional model also has several advantages. With the torsional stiffness model, edge-based torsional spring forces are superimposed onto the lineal spring forces [54], leading to a similar formulation as shown in Eq. (56). However, the semi-torsional approach presented here entails much less computational cost than the torsional stiffness approach. Briefly, in a 2D triangular mesh, 72 additions and 117 multiplications per element are required to compute the torsional stiffness, while only

18 additions and 20 multiplications per element are needed to compute the semi-torsional stiffness.

Extension of the semi-torsional stiffness to a 3D tetrahedral mesh is achieved by defining a facing angle of an edge formed by two faces of a tetrahedron. Semi-torsional stiffness is calculated directly from the coordinates of the four vertices of the tetrahedron, which requires 26 additions and 25 multiplications per element. The cost is slightly higher than that of 2D semi-torsional model, but much lower than that of 2D torsional model. With the torsional approach, extension to 3D necessitates a number of major intermediate steps, including insertion of triangles into a tetrahedron, applying 2D torsional approach on each inserted triangle, interpolation at one vertex of each inserted triangle, transformation from the local frame of each triangle to the 3D frame, etc., through which non-trivial overhead cost is incurred.

Using torsional springs leads to kinematic formulations based on the assumption of small displacements and small rotations [54]. Deformation of the computational domain can be small between two adjacent time steps even when large deformation of the computation domain occurs throughout the whole simulation process. However, this assumption imposes a limit on time step size. In the semi-torsional approach, no such kinematic formulation is needed, hence no assumption on small displacements between two adjacent time steps is made, suggesting that large deformation may be imposed on a computational domain within a single time step.

7.2.4 Boundary improvement

Since the static equilibrium equations for the mesh are elliptic, the principle of Saint Venant holds for deformation of the mesh. Therefore, boundary displacement does not spread far into the mesh. A boundary-improvement technique was suggested to handle this localization of deformation [57]. The stiffness of springs adjacent to the boundary was increased so that surface displacement could be spread further into the mesh. To implement this, the coefficient λ is magnified by a constant factor for springs adjacent to the boundary.

Several layers of elements near the boundary can be made stiffer by this boundary improvement technique. In our experiments, we imposed one layer of boundary stiffness modification by increasing λ from 1.0 to 3.5.

7.2.5 SOR solver

We call two nodes *spring neighbors* if they are directly connected by a fictitious spring. A data structure known as *spring neighbor table* is created, in which spring neighbors of each node are tabulated, as well as the total number of neighbors that a node has and the global node number of each neighbor. Elements sharing each edge are also stored in the spring neighbor table.

Maintaining the spring neighbor table increases memory requirement slightly in a flow simulation, but the data structure greatly facilitates computing lineal stiffness and semi-torsional stiffness when the linearized system of Eq. (46) is iteratively solved. We implemented the successive over-relaxation (SOR) algorithm to solve the linear system for nodal displacements at internal nodes, with the over-relaxation parameter set between 1.35 and 1.4. This range was found to be optimal based on tests on a coarse mesh with 15,669 elements and a fine mesh of 61,079 elements. The iterations at time step n were started from the solution at the previous time step $n-1$, and the system was linearized by lagging the spring constants based on the geometry of the previous time step.

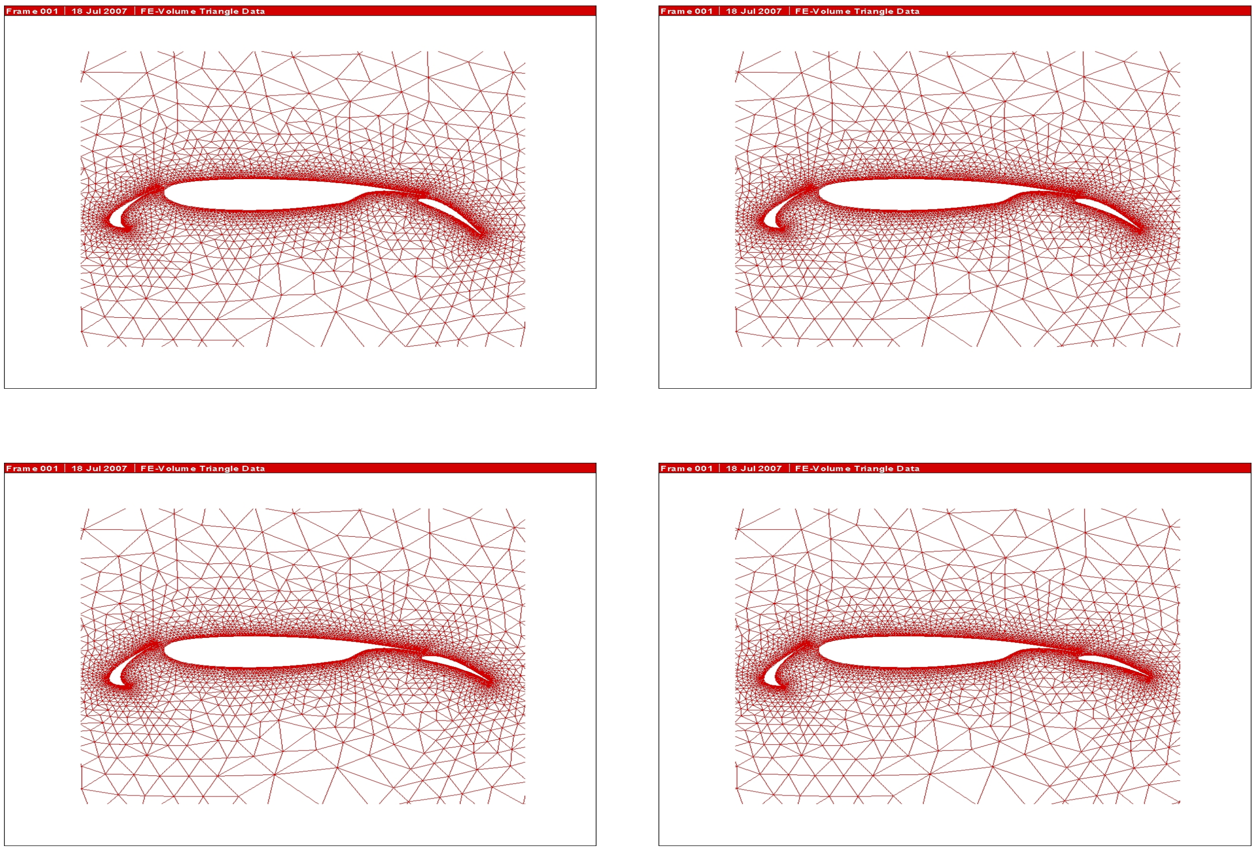
The stopping criterion for the SOR iteration was

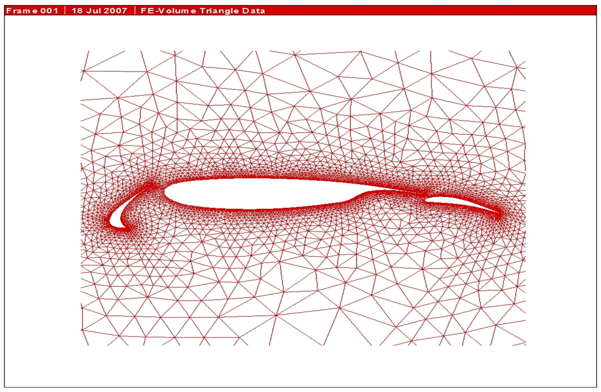
$$\max\left\{\left|\delta_{1,i}^{k+1}-\delta_{1,i}^{k}\right|,\left|\delta_{2,i}^{k+1}-\delta_{2,i}^{k}\right|,\left|\delta_{3,i}^{k+1}-\delta_{3,i}^{k}\right|\right\}\leq\varepsilon, \tag{57}$$

where i ranged over all internal nodes, k denoted iteration, and δ_1, δ_2, δ_3 were three components of nodal displacement. Based on a variety of preliminary tests, we set ε between 5×10^{-7} and 5×10^{-6}, using single precision calculations. This ensured robust convergence at an acceptable computational cost.

7.3 Mesh Testing Results

We implemented the semi-torsional model for 2D multi element airfoil with slat and flap movement, see following Figure 17. In this example, the main-body of airfoil is fixed, positions and angulars of slat and flap are changed. Then, the interior mesh is modified by using above analogy methods according the boundary displacement.


Frame 001 | 18 Jul 2007 | FE-Volume Triangle Data
Frame 001 | 18 Jul 2007 | FE-Volume Triangle Data
Frame 001 | 18 Jul 2007 | FE-Volume Triangle Data
Frame 001 | 18 Jul 2007 | FE-Volume Triangle Data



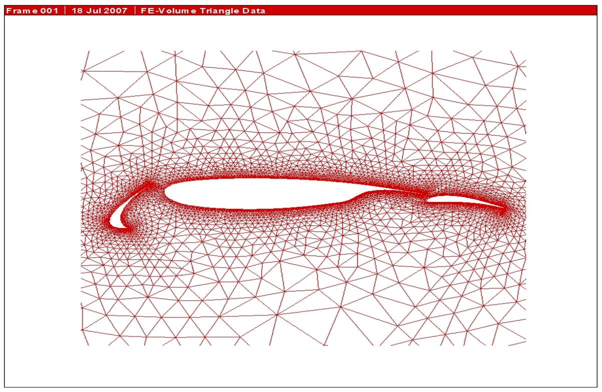


Figure 17: Example of 2D springs methods for unstructured meshes

8 APPLICATION TO HIGH LIFT DEVICE OPTIMIZATION WITH UNCERTAIN ANGLE OF ATTACK

8.1 Optimization Description

All aircraft designs include a number of compromises intended to maximize performance for a particular role. One of the most fundamental of these is the size of the wing; a larger wing will provide more lift and make takeoffs and landings shorter and easier, but increase drag during cruising and thereby lead to lower fuel economy. High-lift devices are used to smooth out the differences between the two goals, allowing the use of an efficient cruising wing, and adding lift for takeoff and landing.

The most common high-lift device is the flap, a movable portion of the rear wing that can be bent down into the airflow to produce extra lift. The effect of the flap is not as obvious as it may seem, the real goal is to re-shape the wing as a whole into one that has more camber as well as being longer. In general wings with more camber and chord will produce more lift for any given amount of drag. It is the second goal, making the wing longer, that results in the complex flap arrangements found on many modern aircraft.

Another common high-lift device is the slat, what appears to be a flap at the front of the wing. In fact the action of the slat is very different than the flap, as it does not directly produce extra lift. Instead the slat re-directs the airflow at the front of the wing, allowing it to flow more smoothly over the surface while at a high angle of attack. This allows the wing to be operated effectively at higher angles, which produce more lift.

The goal of the optimization is to maximize the lift coefficient C_l by modifying the positions and angles of slat anfd flap. The aerodynamic coefficient are computed using N-S flow solver.

8.2 Parameterization

A CAD-free parameterization is used in this optimization, because only the positions and angles of slat and flap can be modified for landing and takeoff fly condition, their shapes and main body is fixed during the optimization. This is due to the global shape (slat + main body + flap) is determined by transonic performance at cruise condition.

Here the design variables are the positions and angles of slat and flap, so totally we have six design variables.

8.3 Single-point Lift Maximization Design at Landing Fly Condition

For landing, we only concern about the maximum lift, because drag is useful for landing. So, the optimization problem is defined as

$$\max C_l \tag{58}$$

The nominal operating condition are defined for landing condition by the free-stream incidence $\alpha = 15^0$, Mach number $M_\infty = 0.15$ and Reynolds number $Re = 1.8 \times 10^6$.

Figure 18 shows the convergence history of lift coefficient. The optimized airfoil slat and flap positions are shown in figure 19 compared with the baseline multi element airfoil, red one shows the optimization airfoil positions and blue one is the baseline airfoil. The optimized pressure distribution is shown in Figure 20, red points show the pressure distribution on optimized airfoil and blue ones show baseline pressure distribution. Table 1 gives the detailed lift coefficient value during optimization.

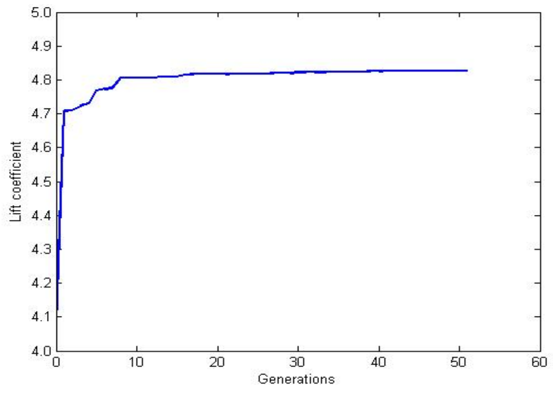


Figure 18: Convergence history of lift coefficient

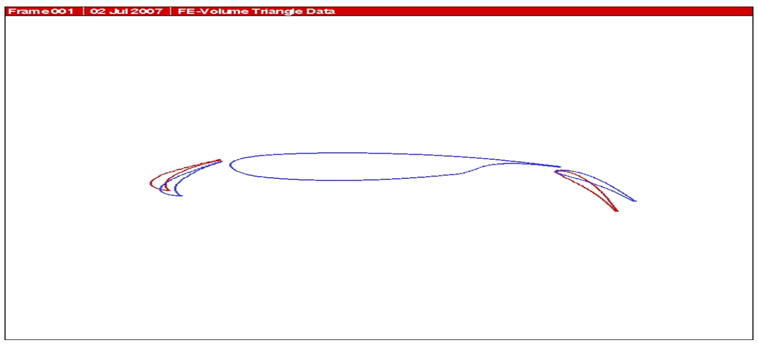


Figure 19: Optimized multi element airfoil configuration for landing

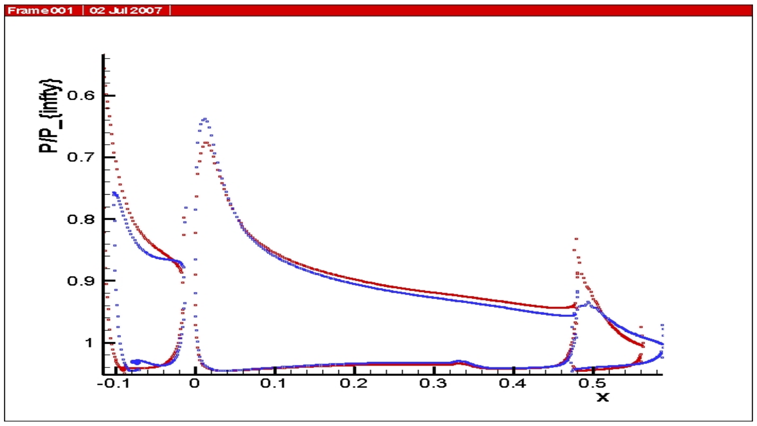


Figure 20: Comparison of pressure distributions between optimized and baseline for landing

Table 1: Lift coefficient values during optimization

Generations	0	5	10	15	20	25	30	35	40	45	50
C_l	4.073	4.768	4.805	4.808	4.816	4.818	4.822	4.823	4.825	4.826	4.827

8.4 Single-point Lift Maximization Design at Takeoff Fly Condition

For takeoff, we concern about not only maximum lift but also minimum drag. So, the optimization problem is defined as

$$\max \frac{C_l}{C_d} \tag{59}$$

The nominal operating condition are defined for takeoff condition by the free-stream incidence $\alpha = 15^0$, Mach number $M_\infty = 0.15$ and Reynolds number $Re = 1.8 \times 10^6$.

Figure 21 shows the convergence history of aspect ratio of lift to drag coefficient. The optimized airfoil slat and flap positions are shown in figure 22 compared with the baseline multi element airfoil, red one shows the optimization airfoil positions and blue one is the baseline airfoil. The optimized pressure distribution is shown in Figure 23, red points show the pressure distribution on optimized airfoil and blue ones show baseline pressure distribution. Table 2 gives the detailed aspect ratio value of lift to drag coefficient during optimization.

Table 2: Aspect ratio values of lift to drag coefficient during optimization

Generations	0	5	10	15	20	25	30	35	40	45	50
C_l/C_d	17.554	24.904	25.304	25.941	26.171	26.292	26.306	26.343	26.351	26.354	26.360

Compare the optimized airfoil configurations of above two optimization problems, we noted that the flap is more slopping down for Landing than for takeoff. This is due to both maximum lift and maximum drag are needed for landing, but maximum lift and minimum

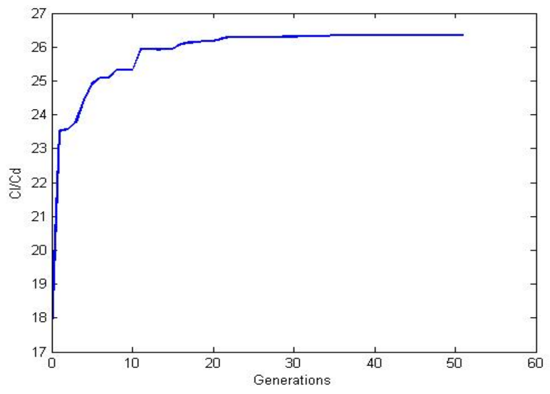


Figure 21: Convergence history of aspect ratio of lift to drag coefficient

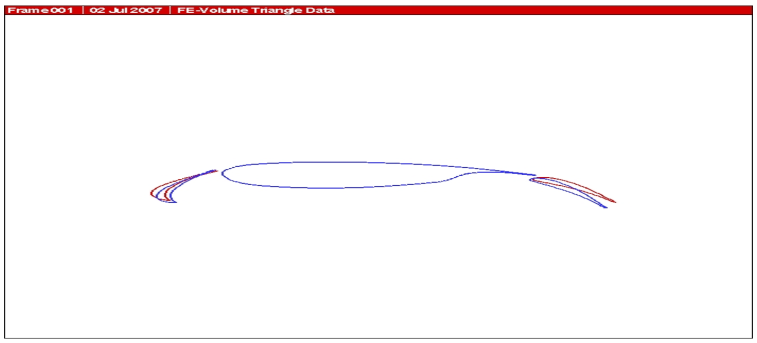


Figure 22: Optimized multi element airfoil configuration for takeoff

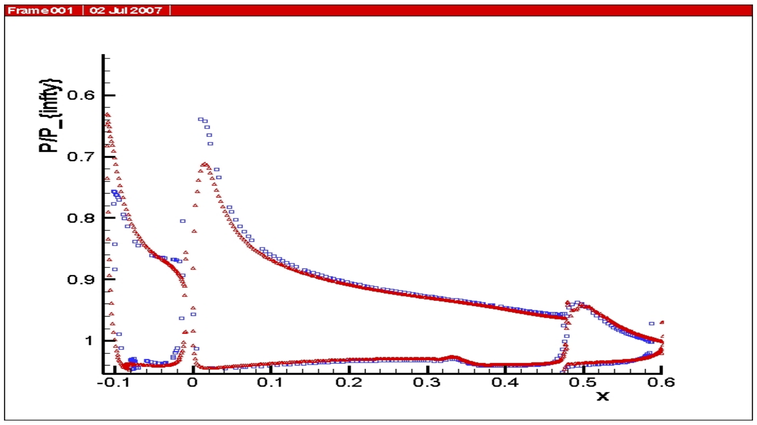


Figure 23: Comparison of pressure distributions between optimized and baseline for takeoff

drag are needed for takeoff. This can be reflected from the final optimized performance of three element airfoil, see table 3.

Table 3: Comparison of optimized airfoil performances with baseline ones

	Optimization Criterion	C_l	C_d
Baseline configuration		4.3757	0.1939
Optimized (Landing)	$\max C_l$	4.8271	0.2328
Optimized (Takeoff)	$\max \frac{C_l}{C_d}$	4.3283	0.1642

8.5 Lift Maximization With Uncertain Angle of Attack at Landing Fly Condition

We suppose that the free-stream angle of attack is subject to random fluctuations. For simplicity, we assume that its PDF is Gaussian with a given mean and variance. The mean angle of attack corresponds to the nominal incidence 15^0 and its standard deviation is 2^0. Free-stream Mach number is $M_\infty = 0.15$ and Reynolds number $Re = 1.8 \times 10^6$. The mathematical formulation of optimization problem defined as

$$\max C_l \qquad at\ \ M_\infty = 0.15,\ \ \alpha = [15^0 - 2^0, 15^0 + 2^0]. \tag{60}$$

According the Taguchi robust control theory, the above design problem with uncertainties can be converted into the following two-objective optimization problem, one objective is the mean value of lift coefficient, the other is the variance of lift coefficient over range of uncertainty.

$$\begin{cases} \max f_1 = \mu_{C_l} = \dfrac{1}{N}\displaystyle\sum_{i=1}^{N} C_{li} \\ \\ \min f_2 = \sigma_{C_l} = \dfrac{1}{N-1}\displaystyle\sum_{i=1}^{N}(C_{li} - \mu_{C_l})^2 \end{cases} \quad over\ \ M_\infty = 0.15,\ \ \alpha_i = [13^0, 14^0, 15^0, 16^0, 17^0], \tag{61}$$

where $N = 5$. Above two-objective optimization problem is solved via GAs by integrating two objectives into single objective, as follow

$$\min f = 10 - f_1 + 100 * f_2 \tag{62}$$

Figure 24 shows the convergence history of integrated fitness function f. The optimized airfoil slat and flap positions are shown in figure 25 compared with the baseline multi element airfoil and traditional optimized airfoil without uncertain fluctuation on angle of attack, red one shows the robust optimized airfoil positions, blue one is the baseline airfoil and green one is traditional designed airfoil in section 8.3. Stream lines over optimized airfoil are shown in

Figure 26, it is obviously that there are two vortices behind the slat and flap. The optimized pressure distribution is shown in Figure 27, red points show the pressure distribution on optimized airfoil and blue ones show baseline pressure distribution.

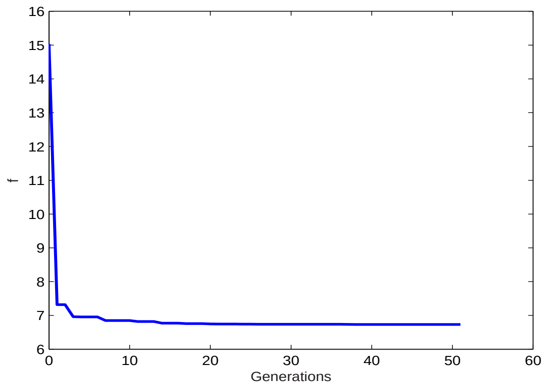


Figure 24: Convergence history of robust lift maximization for landing

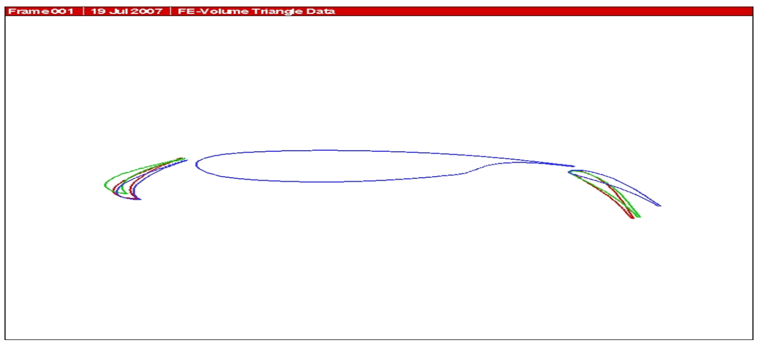


Figure 25: Robust optimized multi element airfoil configuration for Landing

Compare the lift coefficient of robust optimized, traditional single point optimized and baseline airfoil, see Figure 28. We can see that lift coefficient of robust optimized airfoil is not as insensitive as single-point optimized one to the fluctuation on angle of attack, it is also illustrated by the value of C_l of optimized and baseline airfoils in table 4.

9 CONCLUSION

The problem of aerodynamic shape optimization with uncertain operating conditions is addressed in this report. To overcome the difficulty related to the high computational cost required by robust design and GAs, a response surface modelling strategy is proposed, that relies on the polynomial approximation, to estimate the fitness value. A robust optimization problem is then solved by using taguchi concept: converting deign with uncertainties into a two-objective optimization problem, one objective is the mean performance, the other one is the variance of performance.

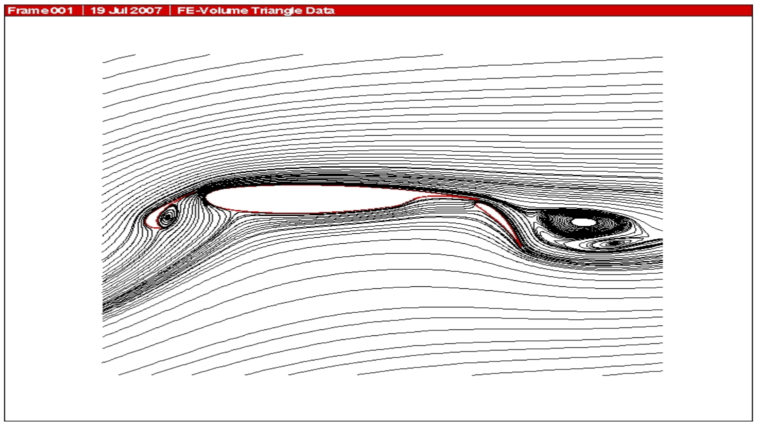


Figure 26: Stream lines over the robust optimized airfoil

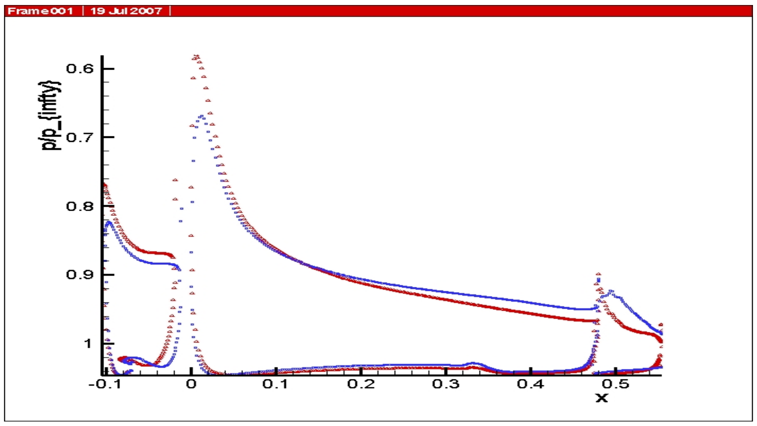


Figure 27: Comparison of pressure distributions between robust optimized and baseline airfoil for landing

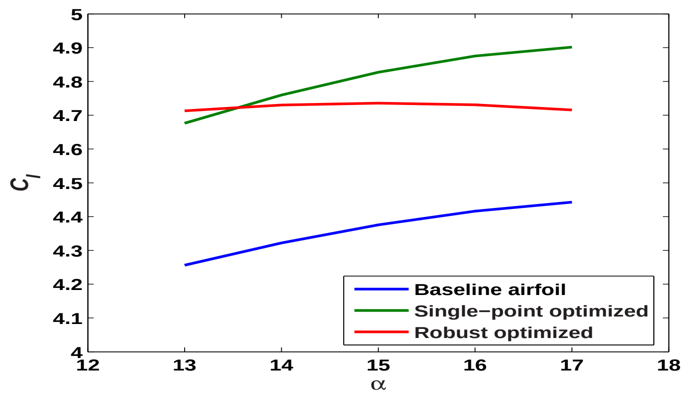


Figure 28: Lift coefficient comparison among robust optimized, traditional single-point optimized and baseline airfoils

Table 4: Comparison of robust optimized airfoil performances with traditional single-point optimized and baseline ones

	α	13^0	14^0	15^0	16^0	17^0
Baseline Airfoil	C_l	4.2461	4.3220	4.3757	4.4161	4.4427
	C_d	0.1742	0.1837	0.1939	0.2041	0.2147
Optimized (Landing)	C_l	4.6762	4.7596	4.8271	4.8753	4.9015
	C_d	0.2130	0.2224	0.2328	0.2458	0.2602
Robust Optimized (Landing)	C_l	4.7129	4.7302	4.7358	4.7307	4.7155
	C_d	0.2440	0.2540	0.2642	0.2751	0.2870

This methodology is demonstrated for a realistic high lift device's lift maximization in subsonic flow with fluctuation on free stream incidence angle. This optimization problem is solved using the proposed Taguchi robust control method successfully.

The proposed approach has been found particularly effective to drive a shape optimization procedure to a robust design.

10 FUTURE WORKS

1. Run the multi-criterion GAs code on a PC cluster to solve the two-objective optimization problem generated by Taguchi robust control methods for design with uncertainties;
2. Use mesh adaptivity technique and prior error estimator idea [64][65][66] into design with uncertainties to achieve sophisticated robust optimization methods;
3. Extend robust design methods addressed in this paper to 3D realistic aircraft designs.

ACKNOWLEDGEMENT

The first author would like to thank the support of CIMNE, which made his visit possible.

The first author would also like to acknowledge E. Oñate, J. Periaux, G. Bugeda, P. A. Ubach and other researchers for many fruitful discussions during his stay in CIMNE/UPC.

Finally the first author is grateful to L. Zielonka and M. Alberich for their help and patience in solving many administrative and logistic problems making his visit to Barcelona not only easier but also enjoyable.

REFERENCES

[1] Drela, M. *Pros and cons of airfoil optimization.* In: Caughey, D.A.; Hafez, M.M. (eds.) Proc. Frontiers of computational fluid dynamics, World Scientific, 1998.

[2] Egorov, I. N.; Kretinin, G. V.; and Leshchenko, I. A. *How to Execute Robust Design.* AIAA Paper No. 2002-5670, 9th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, September 4-6, Atlanta, Georgia, 2002.

[3] Fowlkes, W. Y.; Creveling, C. M. *Engineering methods for robust product design using Taguchi methods in technology and product development.* Reading, MA: Addison-Wesley, 1995.

[4] Ben-Tal, A.; Nemirovski, A. *Robust truss topology design via semidefinite programming.* SIAM J. Optimiz. 7, 991C1016, 1997.

[5] Huyse, L. *Free-form airfoil shape optimization under uncertainty using maximum expected value and secondorder second-moment strategies.* NASA/CR-2001-211020 or ICASE Report No. 2001-18, 2001.

[6] Huyse, L.; Lewis, R. *Aerodynamic shape optimization of two-dimensional airfoils under uncertain operating conditions.* NASA/CR-2001-210648 or ICASE Report No. 2001-1, 2001.

[7] Li, W.; Huyse, L.; Padula, S. *Robust airfoil optimization to achieve consistent drag reduction over a Mach range.* NASA/CR-2001-211042 or ICASE Report No. 2001-22, 2001.

[8] Deb, K. *Multi-Objective Optimization Using Evolutionary Algorithms.* Wiley, 2001.

[9] Gonzalez, L.; Whitney, E.; Srinivas, K. and Periaux, J. *Multidisciplinary Aircraft Design and Optimization Using a Robust Evolutionary Technique with Variable fidelity Models.* AIAA Paper No. 2004- 4625, Tenth AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Albany, New York, August 30 C Sep. 1, 2004.

[10] Langer, H.; Puehlhofer, T. and Baier, H. *A Multiobjective Evolutionary Algorithm with Integrated Response Surface Functionalities for Configuration Optimization with Discrete variables.* AIAA Paper No. 2004-4326, Tenth AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Albany, New York, August 30 C Sep. 1, 2004.

[11] Chung, H. and Alonso, J. *Multiobjective Optimization Using Approximation Model Based Genetic Algorithms.* AIAA Paper No. 2004-4325, Tenth AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Albany, New York, August 30 C Sep. 1, 2004.

[12] Marco N. M.; Désidéri J. A.; Lanteri S. *Multi-objective optimization in CFD by genetic algorithms.* INRIA Report RR-3686, INRIA Sophia Antipolis, 1999.

[13] Michael W. T. *Taguchi and Robust Optimization.* http://citeseer.ist.psu.edu/cache/papers/cs/5981

[14] Demýanov, V. F. and Malozemov, V. N. *Introduction ot Minimax.* Dover Publication, New York, 1974.

[15] Das, I. *Multicriteria Optimization and Robust Optimality.* Thesis proposal, Department of Computational & Applied Mathematics, Rice University, Houston, TX, 1996.

[16] Davis, P. J. and Rabinowitz, P. *Methods of Numerical Integration.* Academic Press, Orlando, FL, 1984.

[17] Flournoy, N. and Tsutakawa, R. K. *Statistical Multiple Integration.* American Mathematical Society, Providence, RI, 1991.

[18] Tierney, L. and Kadane, J. B. *Accurate Approximations for Posterior Moments and Marginal Densities.* Journal of the American Statistical Association, 81:82-86, 1986.

[19] Kass, R. E.; Tierney, L. and Kadane, J. B. *Approximate methods for Assessing Influence and Sensitivity in Bayesian Analysis.* Biometrika, 76: 663-674, 1989.

[20] Wong, W. H. and Li, B. *Laplace Expansion for Posterior Densities of Nonlinear Functions of Parameters.* Biometrika, 79:393-398, 1992.

[21] Roy, R. K. *A Primer on the Taguchi Method.* Competitive Manufacturing. Van Nostrand reinhold, New York, 1990.

[22] nair, V. N. *Taguchi's Parameter Design: A panel discussion.* Technometrics, 34:127-161, 1992.

[23] Michael, W. T.; Natalia M. A.; Layne T. W. *New Methods for Robust Design Using Computer Simulations.* http://www.math.wm.edu/ trosset/Research/DACE/robust2.pdf

[24] Welch, W. J. and Sacks, J. *A system for quality improvement via computer experiments.* Communications in Statistics-Theory and Methods, 20:477-495, 1991.

[25] Ferguson, T. S. *Mathematical Statistics: A Decision Theoretic Approach.* Academic Press, New York, 1967.

[26] Welch, W. J.; Yu, T. K.; Kang, S. M.; and Sacks, J. *Computer experiments for quality con- trol by parameter design.* Journal of Quality Technology, 22:15-22, 1990.

[27] Tang, Z. L.; Périaux, J.; Désidéri, J. A. *Multi Criteria Robust Design Using Adjoint Methods and Game Strategies For Solving Drag Optimization Problems With Uncertainties.* West-East High Speed Flow Field Conference (WEHSFF05), Beijing, October, 19-22, 2005.

[28] Duvigneau, R. *Aerodynamic Shape Optimization with Uncertain Operating Conditions using Metamodels*. INRIA report RR-6143, 2007.

[29] Zingg, D. and Elias, S. *Aerodynamic Optimization Under a range of operating conditions*. AIAA Journal. 44, 11, 2787-2791, 2006.

[30] Walter, R. and Huyse, L. *Uncertainty analysis for fluid mechanics with application*. Tech. Rep. 2002-1, ICASE, February, 2002.

[31] Jin, R.; Du, X. and Chen, W. *The use of metamodeling techniques for optimization under uncertainty*. In ***ASME*** Design Engineering Technical Conferences, September 9-11, Pittsburgh, USA, 2001.

[32] Darwin, C. *On the origin of species by means of natural selection*. London, John Murray, 1859 [1st edn.].

[33] Marco, N. and Désidéri, J.A. *Numerical solution of optimization test-case by Genetic Algorithms*. INRIA Research Report no. 3622, February 1999.

[34] Marco, N.; Lanteri, S. *A Two-Level Parallelization Strategy for Genetic Algorithms Applied to Shape Optimum Design*. INRIA Research Report no. 3463, July 1998.

[35] Mäkinen, R. A.; Toivanen, J. and Periaux, J. *Multidisciplinary Shape Optimization in Aerodynamics and Electromagnetics using Genetic Algorithms*, International Journal for Numerical Methods in Fluids, Volume 30, Issue 2, Pages 149- 159, 1999.

[36] Schaffer, J. D. *Some experiments in machine learning using vector evaluated genetic algorithms*. TCGA file no.00314, Ph.D thesis, Vanderbilt University, Nashville, TN, 1984.

[37] Srinivas, N. and Dep, K. *Multiobjective optimization using nondominated sorting in genetic algorithms*. Evolutionary Computing, 3, 221-248, 1995.

[38] Fonseca, C. M. and Fleming, P.J. *Multiobjective Optimizers*. in H.-M. Voigt, W. Ebeling, I. Rechenberg and H.-P. Schwefel (eds), Parallel Problem Solving from Nature - PPSN IV, International Conference on Evolutionary Computation, Vol. 1141, Lecture Notes in Computer Science, Springer, Berlin, 584-593, 1996.

[39] Mäkinen, R. A. E. and Toivanen, J. *Parallel solution of optimal shape design problem governed by Helmholtz/potential flow equations*. in D. H. Bailey et al. (eds), The 7th SIAM Conference on Parallel Processing for Scientific Computing, SIAM, 102-103, 1995.

[40] Michalewicz, Z. *Genetic algorithms + data structures = evolution programs*. Springer-Verlag, 1992.

[41] Périaux, J.; Sefrioui, M.; Stouflet, B.; Mantel, B.; Laporte, E. *Robust genetic algorithms for optimization problems in aerodynamic design*. Genetic algorithms in engineering and computer science, John Wiley & Sons, pp. 397-415, 1995.

[42] Quagliarella, D. *Genetic algorithms applications in computational uid dynamics, Genetic algorithms in engineering and computer science.* John Wiley & Sons, pp. 417-442 1995.

[43] Spendley, W.; Hext, GR.; Himsworth, FR. *Sequential application of simplex designs in optimization and evolutionary operation.* Technometrics 4 (1962), 441-461, 1962.

[44] Giunta, A. A.; Narducci, R.; Burgee, S.; Grossman, B.; Mason, W. H.; Watson, L. T.; and Haftka, R. T. *Variable-Complexity Response Surface Aerodynamic Design of an HSCT Wing.* in Proceedings of the 13th AIAA Applied Aerodynamics Conference, pp. 994-1002, San Diego, CA, AIAA Paper 95-1886, June, 1995.

[45] Gilmore, P. and Kelley, C. T. *An Implicit Filtering Algorithm for Optimization of Functions with Many Local Minima.* SIAM J. Opt., 5(2), 269-285, 1995.

[46] Zeng, D.; Ethiera, C. R. *A semi-torsional spring analogy model for updating unstructured meshes in 3D moving domains.* Finite Elements in Analysis and Design, Vol. 41, pp1118-1139, 2005.

[47] Hirt, C. W.; Nichols, B. D. *Volume of fluid (Vof) method for the dynamics of free boundaries.* J. Comput. Phys. 39, 201, 1981.

[48] Douglass, R. W.; Carey, G. F.; White, D. R.; Hansen, G. A.; Kallinderis, Y.; Weatherill, N. P. *Current views on grid generation: summaries of a panel discussion.* Numer. Heat Transfer Part BFund. 41. 211., 2002.

[49] Samareh, J. A. *Status and future of geometry modeling and grid generation for design and optimization.* J. Aircraft 36. 97. 1999.

[50] Teng, S. H.; Wong, C. W. *Unstructured mesh generation: theory, practice, and perspectives.* Int. J. Comput. Geom. Appl. 10. 227., 2000.

[51] Batina, J. T. *Unsteady Euler airfoil solutions using unstructured dynamic meshes.* AIAA J. 28. 1381, 1990.

[52] Blom, F. J. *Considerations on the spring analogy.* Int. J. Numer. Methods Fluids 32. 647, 2000.

[53] Degand, C.; Farhat, C. *A three-dimensional torsional spring analogy method for unstructured dynamic meshes.* Comput. Struct. 80. 305, 2002.

[54] Farhat, C.; Degand, C.; Koobus, B.; Lesoinne, M. *Torsional springs for two-dimensional dynamic unstructured fluid meshes.* Comput. Methods Appl. Mech. Eng. 163. 231, 1998.

[55] Huerta, A.; Liu, W. K. *Viscous flow with large free surface motion.* Comput. Methods Appl. Mech. Eng. 69. 277, 1988.

[56] Bartels, R. E. *Mesh strategies for accurate computation of unsteady spoiler and aeroelastic problems*. J. Aircraft 37. 521, 2000.

[57] Blom, F. J.; Leyland, P. *Analysis of fluid-structure interaction by means of dynamic unstructured meshes*. J. Fluids Eng. -Trans. ASME 120. 792, 1998.

[58] Hassan, O.; Probert, E. J.; Morgan, K. *Unstructured mesh procedures for the simulation of three-dimensional transient compressible inviscid flows with moving boundary components*. Int. J. Numer. Methods Fluids. 27. 41, 1998.

[59] Lomtev, I.; Kirby, R. M.; Karniadakis, G. E. *A discontinuous GalerkinALE method for compressible viscous flows in moving domains*. J. Comput. Phys. 155. 128, 1998.

[60] Piperno, S. *Explicit/implicit fluid/structure staggered procedures with a structural predictor and fluid subcycling for 2D inviscid aeroelastic simulations*. Int. J. Numer. Methods Fluids. 25. 1207, 1997.

[61] Rausch, R. D.; Batina, J. T.; Yang, H. T. Y. *3-dimensional time-marching aeroelastic analyses using an unstructured-grid Euler method*. AIAA J. 31. 1626, 1993.

[62] Tsai, H. M.; Wong, A. S. F.; Cai, J.; Zhu, Y.; Liu, F. *Unsteady flow calculations with a parallel multiblock moving mesh algorithm*. AIAA J. 39. 1021, 2001.

[63] Zeng, D.; Ethier, C.R. *A semi-torsional spring analogy model for updating unstructured meshes*. in: G.E. Schneider (Ed.), Proceedings of the Ninth Annual Conference of the CFD Society of Canada, Water loo, Ont., Canada, 5-27-2001, vol. 1, The CFD Society of Canada, 5-27-2001, p. 113.

[64] Bugeda, G.; Oñate, E. *Optimum Aerodynamic Shape Design Including Mesh Adatpivity.* Int. J. Num. Meth. in Fluid, Vol. 20, pp915-934, 1995.

[65] Bugeda, G.; Oñate, E. *Optimum Aerodynamic Shape Design for Fluid Flow Problems Including Mesh Adaptivity.* Int. J. Num. Meth. in Fluid, Vol. 30,161-178, 1999.

[66] Bugeda, G.; Ródenas, J. J.; Oñate, E. *An Integration of a Low Cost Adaptive remeshing Strategy in The Solution of Structural Shape Optimization Problems Using Evolutionary Methods*. Interior Communication, 2006.