An Adaptive Multipoint Formulation for Robust Parametric Optimization

Francois Gallard, Bijan Mohammadi, Marc Montagnac, Matthieu Meaux

An Adaptive Multipoint Formulation for Robust Parametric Optimization

François Gallard · Bijan Mohammadi · Marc Montagnac · Matthieu Meaux

Abstract The performance of a system designed for given functioning conditions often seriously degrades, when operated at other conditions. Therefore, a system operating over a continuous range of conditions should be designed over this range. The aerodynamic shape optimization of an aircraft at multiple altitudes, angles of attack and Mach numbers is a typical case in aerospace. This paper links parametric and multipoint optimizations by the sampling of the operating condition ranges. It is demonstrated that this discrete set of operating conditions, used to formulate a composite objective function, must adequately be chosen. An algorithm is proposed to select these conditions, which ensures a minimal computational cost to the robust optimization. Wing aerodynamic multipoint optimizations using a lifting line model and Reynolds-averaged Navier–Stokes equations, derived with a discrete adjoint formulation, are given as examples.

Communicated by Emmanuel Trélat.

F. Gallard (✉)
Institute of Technology Antoine de Saint Exupéry, 118 route de Narbonne,
CS 44248, 31432 Toulouse Cedex 4, France
e-mail: francois.gallard@irt-saintexupery.com

M. Montagnac
Computational Fluid Dynamics, Centre Européen de Recherche et de Formation Avancée
en Calcul Scientifique, 42, Avenue Gaspard Coriolis, 31057 Toulouse Cedex 1, France
e-mail: marc.montagnac@cerfacs.fr

B. Mohammadi
Université de Montpellier, Mathématiques (IMAG), CC51, 34095 Montpellier, France
e-mail: Bijan.Mohammadi@umontpellier.fr

M. Meaux
Airbus Group Innovations, 18 rue Marius TERCE, BP 13050, 31025 Toulouse Cedex 03, France
e-mail: matthieu.meaux@airbus.com

Keywords Parametric optimization · Multipoint optimization · Gradient-based optimization · Robust optimization · Aerodynamics · Adjoint method

Mathematica Subject Classification 93B51 · 35Q93 · 49M37 · 49Q10 · 90C31

1 Introduction

In most practical situations, the cost function of an optimization problem depends not only on design variables but also on extra parameters. A typical case is when the performance of a system must be improved over a range of operating conditions, each operating condition being represented by a set of parameters. This type of problem is called parametric optimization in the following, and aims at finding the design variables that give the best user-defined performance over this range. Designers are in charge of defining the performance of an aircraft. In other words, they must carefully specify the optimization problem and the trade-offs between all constraints and operating conditions as they greatly affect the overall objective, and thus the final optimum.

During a mission, an aircraft flies under various operating conditions such as the angle of attack, the altitude, and the Mach number. The aerodynamic shape design must then take them into account in the optimization process since optimizing at only one condition is known to lead to poor off-design performance, which has been reported as drag-creep [1], single-point optimization effect [2], or localized optimization [3]. The single-point optimization effect is also undesirable, since the condition chosen to perform the optimization may not exactly be the real flight condition.

The term robust has many meanings as mentioned in [4]. Here, it is important to note that no randomness is involved in the formulation of our problem. Besides, the problem is different when optimization takes into account manufacturing or design variable uncertainties. The present methodology addresses the optimization of deterministic design variables, when the same product needs to perform well in a range of operating conditions.

Therefore, either the objective function or the constraints must take into account the operating conditions in the formulation of the optimization problem. This class of problems is classically tackled with multipoint optimization methods in the literature [5–10], just to cite a few references from the aerodynamic community.

A set of functions to be minimized, coming from a set of operating conditions, can be seen as a set of concurrent objectives, and the problem is then in the scope of multi-objective optimization. Multi-objective algorithms, such as Normal Boundary Intersection [11], Normal Constraint [12], Successive Pareto Optimization [13], Directed Search Domain [14], and Multiple-Gradient Descent Algorithm [15], aim at obtaining the Pareto front. Genetic algorithms, handling populations of solutions, can also provide Pareto fronts. For instance, the Covariance Matrix Adaptation Evolution Strategy [16] and Non-dominated Sorting Genetic Algorithm-II [17] algorithms have been used in the aerodynamic community. Marco et al. [18] performed Euler airfoil optimization using genetic algorithms. Vicini and Quagliarella [19] solved inverse and direct airfoil design optimization problems using a multi-objective genetic algo-

rithm. Another alternative, combining Nash games and gradient algorithms, was proposed by Tang et al. [20], who achieved two-objective airfoil optimizations in Euler conditions.

When the preference of the user among multiple solutions is hard to define a priori, there is a clear interest in obtaining the Pareto front. However, in our case, a designer delivers a single solution and not a set of solutions, since a single system is built. So a decision has to be made at some point, based on quantitative criteria. When such a criterion can be expressed before the optimization, it is advantageous to take it into account in the problem formulation, since multi-objective algorithms require more function evaluations than the single-objective ones. This aspect is accentuated when the objectives, such as solutions of 3D numerical simulations of viscous flows, are expensive to compute. For instance, the computational cost to obtain the Pareto front of a 2D airfoil optimization with two objectives was 57 times higher than the cost of a single-objective optimization using a gradient algorithm in [21]. In another example of airfoil optimization [20], from 160 to 600 gradient optimizations were required to obtain the Pareto front, depending on the case, with two objectives. When both the number of objectives and the CPU cost of the function evaluation increase, the cost of multi-objective algorithms can become unaffordable. This is the case of the Navier–Stokes 3D optimization given as illustration at the end of the present paper. It has 6 objectives, and the evaluation of a single-objective function with gradients costs 500 CPU hours, which is more than $10,000$ times higher than the function evaluation on a 2D Euler airfoil case.

Besides, as shown in [22] and [23], for multipoint aircraft aerodynamic optimization cases, there exists a user preference function that aggregates the objectives: the integrated aircraft fuel consumption on a mission. From the gradient of this function, a weighted sum of objectives can be expressed. In [23], a moderate cost estimation of the utopia point is proposed, which also enables to specify an adequate set of weights [24]. Alternatively, Giannessi et al. [25] proposed a weight-free method that aims at minimizing a scalar function, here typically the user preference function, over the Pareto set, without preliminarily finding it.

Second, the minimum of a weighted sum with positive weights is always Pareto-optimal [26], which is a key point from the user's point of view. Some points of the Pareto front may not be reached by the weighted sum method [27]. However, multiple studies suggest that the gradient method is able to reach the Pareto front formed by multipoint aircraft wing and airfoil drag minimization in cruise conditions [21,28,29].

Therefore, we focus in the sequel on methods with a priori articulations of preferences, and in particular on the weighted sum method. In most of the studies using a weighted sum approach, the set of operating conditions is a given input of the optimization problem, and the impact of the choice of these conditions on the design problem is rarely discussed. In [10], authors proposed an iterative heuristic method that detects the drag–creep after each optimization, and that updates the set of operating conditions for the next optimization. We propose a priori method, relying on theorems, for the optimization problem setup in order to prevent the drag–creep effect.

It has been shown in [2] that for transonic flows, even considering many operating conditions, the optimizer can end up with a solution that behaves badly away from these sampled design points. Indeed, the flow is very sensitive to shape deformations in transonic conditions. So, the main drawbacks of the multipoint approach remain the selection of appropriate design points and the choice of the associated weights [27].

This paper aims at linking parametric and multipoint optimizations together with a new sampling method of the operating condition space. The focus is made on the choice of the operating conditions to be incorporated in a scalarized objective, based on the analysis of the gradients of the objective function with respect to the design variables computed at these conditions. The general optimization strategy is presented in Sect. 2, followed by Sect. 3, where two theorems provide the fundamental basis for practical applications. In Sects. 4 and 5, it is shown how to properly build an optimal set of operating conditions so that the performance of the system is controlled over the whole range of these conditions. Applications to wing design on a range of angles of attack and Mach numbers are finally presented in Sects. 6 and 7.

2 Multipoint Objective Function

Given the parameter $\alpha \in \mathbf{I} \subseteq \mathbb{R}$, the typical single-point aerodynamic optimization problem is to find the vector of design variables $x \in \mathbf{O}_{\text{ad}} \subseteq \mathbb{R}^n$ that minimizes the function $\mathcal{J}(U, x, \alpha)$, with $\mathcal{J} \in C^1(\mathbb{R}^{\text{ndf}} \times \mathbb{R}^n \times \mathbf{I}, \mathbb{R})$, where ndf is the number of degrees of freedom of the simulation, subject to the equality constraint $R(U, x, \alpha) = 0$, with $R \in C^1(\mathbb{R}^{\text{ndf}} \times \mathbb{R}^n \times \mathbf{I}, \mathbb{R}^{\text{ndf}})$. U is a solution of the discretized non-linear steady-state flow equations. Design variables come from the parameterization of the aerodynamic shape. The objective function is typically the drag. Fluid dynamics Eq. (1) are Euler or Reynolds-averaged Navier–Stokes (RANS) equations. U represents the flow variables. In aerodynamics, the parameter α is a specified operating condition such as the Mach number, Reynolds number, or angle of attack.

$$R(U, x, \alpha) = 0. \tag{1}$$

The objective function can also be written

$$j(\alpha, x) := \mathcal{J}(U, x, \alpha) \text{ such that } R(U, x, \alpha) = 0. \tag{2}$$

$x_L \leq x \leq x_U$, where x_L and x_U are specified bounds. In practice, j cannot be obtained explicitly, but its value can be computed numerically, as well as its gradients, using Lagrange multipliers.

When α is no longer fixed, but varies continuously in the set $\mathbf{I}$, then the optimization problem can no longer be formulated as previously because $j(x, \alpha)$ is a function of α for a given x. R is now required to be a C^1 function of x, U, and α.

The approach presented in this paper uses the inherent properties of the state Eq. (1) to adequately sample the set $\mathbf{I}$ of operating conditions. Given m parameters, $(\alpha_1, \ldots, \alpha_m) \in \mathbf{I}^m$, and the associated weights, $(\omega_1, \ldots, \omega_m) \in \mathbb{R}^m$, the problem is

to find the vector of design variables $x \in \mathbf{O}_{\text{ad}} \subseteq \mathbb{R}^n$ that minimizes the function,

$$J(x) := \sum_{k=1}^{m} \omega_k \; j(x, \alpha_k). \tag{3}$$

This strategy is similar to the proper orthogonal decomposition (POD) method that has been developed first by Kosambi [30], and then applied to aerodynamics in [31–33]. POD takes as input a set of solutions of the state equation obtained from various operating conditions for instance. The basis for this set of solution vectors is computed using a singular value decomposition. Then, a reduced model is built by selecting the highest singular values. This projection can be used for the approximation of the states when the operating conditions vary. In the present case, the aim is not to approximate the solution of state equations, but to take advantage of the linear dependencies of the set made of the functional gradients with respect to design variables. In both cases, the underlying idea is that there exists an appropriate lower dimension approximation of the system states for the purpose of interpolation or gradient computation.

The choice of the set of conditions is discussed in Sect. 3. Specifying the weights relies on multi-objective optimization analyses. This question is addressed for instance by methods, such as Normal Constraint [12], Successive Pareto Optimization [13], Directed Search Domain [14], and Multiple-Gradient Descent Algorithm [15,20], and is not treated in the present study. In the final example of application, a method to approximate the utopia point based on a physical analysis is used to compute the weights; more details are given in [23].

Composite objective functions (3) show a natural parallelism through the possible independent resolutions of the state equation R and evaluations of the function j that represent the major part of the computational effort. This is sketched in algorithm 1, classically used with variants on the minimization step for a given composite objective gradient [5–10]. The main steps consist in solving the state equations for all the operating conditions, computing the objectives $j(x, \alpha_k)$ and their gradients with respect to the design variables x, aggregating the function and the gradients, and using a gradient-based minimization algorithm to update the design variables.

Algorithm 1: Scheme showing the parallel property of the multipoint optimization Let $\{\alpha_1, \ldots, \alpha_m\} \in \mathbf{I}^m$ be a set of operating conditions, $(\omega_1, \ldots, \omega_m) \in \mathbb{R}^m$ be a set of weights, $\epsilon_A \in \mathbb{R}$, $\epsilon_A > 0$ be a stopping criterion, and $x_0 \in \mathbf{O}_{\text{ad}}$ an initial guess for the design variables.

Step 0. Set the optimization iteration index $l = 1$.

Step 1. Solve the m state equations $R(U_k, x_l, \alpha_k) = 0$.

Step 2. Perform m parallel evaluations of $j(x_l, \alpha_k)$.

Step 3. Perform m parallel evaluations of $\nabla_x j(x_l, \alpha_k)$.

Step 4. Define the descent direction $d_l := -\sum_{k=1}^{m} \omega_k \nabla_x j(x_l, \alpha_k)$.

Step 5. If $\|d_l\| < \epsilon_A$, stop; otherwise find $\rho_l^{opt} := \text{argmin}_{\rho \in \mathbb{R}_*^+} \{J(x_l + \rho d)\}$

Step 6. Set $x_{l+1} = x_l + \rho_l^{opt} d$, $l = l + 1$ and go to Step 1.

A general background is now defined, and the next sections focus on the specific issues raised by the parameters $(\alpha_1, \ldots, \alpha_m)$. In particular, a link is made between these parameters and some properties of the objective function (3).

3 Choice of a Set of Conditions

In this section, the minimization of parametric objective functions that depend on a set of operating conditions in addition to design variables is formulated with the weighted sum method (3).

The objective is to control the performance $j(x, \alpha)$ over a range $\mathbf{I}$ of operating conditions, while only a finite number m of conditions are explicitly used in (3). The underlying idea to achieve this goal is that the state equation of the system (1) links the different values $j(x, \alpha)$ computed at each operating condition $\alpha \in \mathbf{I}$. In the aircraft design example, the Navier–Stokes equations are solved for each operating condition α to compute the flow around the same shape defined by x.

Any two conditions can be either cooperative, independent, or concurrent. Modifying the shape and improving the performance at one of them leads to either an improvement, a degradation, or no modification of the performance of the other condition. This point can be exploited to generate the minimal set of conditions α_k that gives the shape design with the same performance as if the whole set $\mathbf{I}$ was considered. Reasoning on the set formed by the gradients of the objective function with respect to the design variables at each sampled condition, Li et al. [3] established that the required number of conditions m in the multipoint optimization is linked to the dimension of the design space n. From now on and throughout the rest of the paper, the term gradient will mean: gradient of the performance function with respect to the design variables.

To summarize [3], a parametric function $j(x, \alpha)$, $\alpha \in \mathbf{I}$, is adequately controlled by a composite objective function $J(x) = \sum_{k=1}^{m} \omega_k \, j(x, \alpha_k)$ over the whole set $\mathbf{I}$ of operating conditions, iff no new descent direction of $j(x, \alpha)$ can be found when α varies in $\mathbf{I}$ compared to the descent directions given by $-\nabla_x j(x, \alpha)$ when α varies in $\{\alpha_1, \ldots, \alpha_m\}$. Because the gradient at each condition is a vector of dimension n, it is stated in [3] that taking at least $m \geq n$ is a necessary condition. At convergence of the gradient-based algorithm to a local minimum, the sum of the weighted gradients is zero in unconstrained cases, so the gradients are linearly dependent. If operating conditions are chosen such that the gradients are linearly independent at the initial guess, this tends to provide poor solutions. As a consequence, at least $n+1$ conditions are taken in practice as expressed by

$$m \geq n + 1. \tag{4}$$

Also, an heuristic method with variable weights is proposed in [3] to avoid this costly dependence to the number of design variables.

Here, new necessary and sufficient conditions are given for controlling the shape performance on $\mathbf{I}$, and they are less restrictive than the condition (4) that appears consequently to be neither necessary nor sufficient. These theorems provide the basis for a selection strategy for the condition set $\mathbf{I_m}$.

3.1 Geometrical Approach of the Gradient Span

As an example, a lift-to-drag polar as a function of the lift coefficient is plotted in Fig. 1. The optimization of this polar belongs to the class of problems with multiple

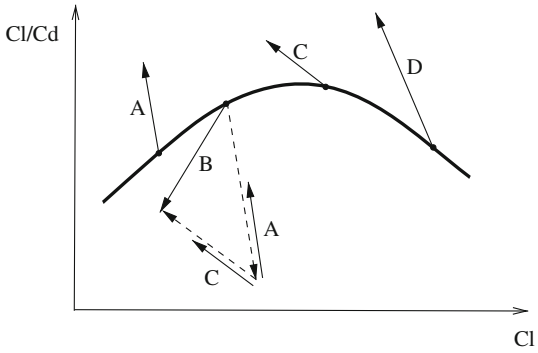


Fig. 1 Polar of lift-to-drag coefficient/lift coefficient and the gradients at four conditions in a design space of dimension two

operating conditions. If two design parameters are considered, the gradients are then in the plane $\mathbb{R}^2$. Thus, Fig. 1 also shows the gradient vectors of four conditions named A, B, C, and D. A basic graphical approach shows that if two gradients are linearly independent as the ones in A and C, then any other gradient, such as the one of the condition B, can be linearly decomposed on the previous ones. This is due to the fact that the set formed by the gradients at conditions A and C is a basis of the plane $\mathbb{R}^2$. So, using the optimization algorithm with the contribution of the gradient at the condition B or of its linear decomposition on the other conditions is identical.

In a general context with n design variables, the design space that contains any gradient is $\mathbb{R}^n$. The point is that it may be impossible to build a basis of $\mathbb{R}^n$ from the gradients computed at all the conditions in $\mathbf{I}$. When the set of gradients from all the conditions is a basis of $\mathbb{R}^n$, the condition (4) becomes effective, and it is our worst case. For that reason, we should focus on the vector space spanned by all descent directions of the objective when the operating conditions vary. In fact, it is sufficient to select a subset of gradients that is a basis of this vector space to setup the optimization problem, as this will be demonstrated in the next sections.

3.2 Implicit Additional Operating Conditions

This section states a basic theorem on the equivalence between the addition of a new operating condition in an initial composite objective function, and the modification of the weights of this function. To the authors' knowledge, this theorem is an original result compared to the state of the art.

Some basic definitions and notations are introduced or recalled first. The functional $j : \mathbb{R}^n \times \mathbf{I} \to \mathbb{R}$ is supposed to be once continuously differentiable. For the sake of simplicity, we consider only one operating condition variable and $\mathbf{I} \subseteq \mathbb{R}$, but the analysis can be extended to more than one dimension. Let $\mathbf{K}_{I,x} := \text{span}\{\nabla_x j(x,\alpha),\ \forall \alpha \in \mathbf{I}\}$ denote the span of the set $\{\nabla_x j(x,\alpha)\}$ of gradient vectors obtained for all $\alpha \in \mathbf{I}$, and $x \in \mathbb{R}^n$. We define $\mathbf{I_m} := \{\alpha_1, \ldots, \alpha_m\}$, a subset of $\mathbf{I}^m$. Let $\mathbf{K}_{I_m,x} := \text{span}\{\nabla_x j(x,\alpha),\ \forall \alpha \in \mathbf{I_m}\}$ denote the span of the subset of $\mathbf{K}_{I,x}$ consisting of the m gradient vectors obtained from $\alpha \in \mathbf{I_m}$.

Let J_{m+1} and J_m be two composite functionals defined by $J_{m+1}(x) := \sum_{k=1}^{m+1} \omega_k j(x, \alpha_k)$ and $J_m(x) := \sum_{k=1}^{m} \bar{\omega}_k j(x, \alpha_k)$, with $\alpha_k \in \mathbf{I_{m+1}}$, $(\omega_1, \ldots, \omega_{m+1}) \in \mathbb{R}^{m+1}$ and $(\bar{\omega}_1, \ldots, \bar{\omega}_m) \in \mathbb{R}^m$.

Now, we analyze the relations between the two vector spaces $\mathbf{K}_{I_m,x}$ and $\mathbf{K}_{I,x}$, and in particular, the equivalence between the following conditions.

Condition 3.1 *$\mathbf{K}_{I_m,x}$ is equal to $\mathbf{K}_{I,x}$.*

Condition 3.2 *For $\alpha_{m+1} \in \mathbf{I}$, $\omega_{m+1} \in \mathbb{R}^*$, and $(\omega_1, \ldots, \omega_m) \in \mathbb{R}^m$, there exists $(\bar{\omega}_1, \ldots, \bar{\omega}_m) \in \mathbb{R}^m$ such that $\sum_{k=1}^{m+1} \omega_k \nabla_x j(x, \alpha_k) = \sum_{k=1}^{m} \bar{\omega}_k \nabla_x j(x, \alpha_k)$.*

Condition 3.3 *For $(\omega_1, \ldots, \omega_m) \in \mathbb{R}^m$, the condition $\sum_{k=1}^{m} \omega_k \nabla_x j(x, \alpha_k) = 0$ implies that, for $\alpha_{m+1} \in \mathbf{I}$, there exists $(\bar{\omega}_1, \ldots, \bar{\omega}_{m+1}) \in (\mathbb{R}^m \times \mathbb{R}^*)$ such that, for all $D \in \mathbb{R}^n$ and $t \in \mathbb{R}$, we have $\sum_{k=1}^{m+1} \bar{\omega}_k\ j(x + t\ D, \alpha_k) = \sum_{k=1}^{m+1} \bar{\omega}_k\ j(x, \alpha_k) + \mathbf{O}(t^2)$.*

Condition 3.4 *For $\alpha \in \mathbf{I}$, we have $\frac{\partial \nabla_x j(x,\alpha)}{\partial \alpha} \in \mathbf{K}_{I_m,x}$.*

Theorem 3.1 *If $j \in C^1(\mathbb{R}^n \times \mathbf{I}, \mathbb{R})$, then Conditions 3.1 and 3.2 are equivalent.*

Proof Let us first prove that Condition 3.1 $\Rightarrow$ Condition 3.2. As $\alpha_{m+1} \in \mathbf{I}$, by definition of $\mathbf{K}_{I,x}$, we have $\omega_{m+1} \nabla_x j(x, \alpha_{m+1}) \in \mathbf{K}_{I,x}$. And, by hypothesis, $\mathbf{K}_{I_m,x} = \mathbf{K}_{I,x}$, then $\omega_{m+1} \nabla_x j(x, \alpha_{m+1}) \in \mathbf{K}_{I_m,x}$. As a consequence, there exists $(\rho_1, \ldots, \rho_m) \in \mathbb{R}^m$ such that $\omega_{m+1} \nabla_x j(x, \alpha_{m+1}) = \sum_{k=1}^{m} \rho_k \nabla_x j(x, \alpha_k)$. Using the latter expression with the following decomposition, $\sum_{k=1}^{m+1} \omega_k \nabla_x j(x, \alpha_k) = \sum_{k=1}^{m} \omega_k \nabla_x j(x, \alpha_k) + \omega_{m+1} \nabla_x j(x, \alpha_{m+1})$, gives $\sum_{k=1}^{m+1} \omega_k\ \nabla_x j(x, \alpha_k) = \sum_{k=1}^{m} (\omega_k + \rho_k)\ \nabla_x j(x, \alpha_k)$. Taking $\bar{\omega}_k = \omega_k + \rho_k$ gives Condition 3.2.

Let us prove now that Condition 3.2 $\Rightarrow$ Condition 3.1. With the hypothesis of Condition 3.2, $\sum_{k=1}^{m+1} \omega_k \nabla_x j(x, \alpha_k) = \sum_{k=1}^{m} \bar{\omega}_k \nabla_x j(x, \alpha_k)$, which is true for any $\omega_{m+1} \in \mathbb{R}^*$. So, for $\omega_{m+1} = 1$, and for any $\bar{\alpha} = \alpha_{m+1} \in \mathbf{I}$, we can write $\nabla_x j(x, \bar{\alpha}) = \sum_{k=1}^{m} (\bar{\omega}_k - \omega_k)\ \nabla_x j(x, \alpha_k)$.

By definition of a span, recall that

$$\mathbf{K}_{I,x} := \{\sum_{i=1}^{p} \lambda_i \nabla_x j(x, \bar{\alpha}_i) \mid p \in \mathbb{N}, (\lambda_1, \ldots, \lambda_p) \in \mathbb{R}^p, (\bar{\alpha}_1, \ldots, \bar{\alpha}_p) \in \mathbf{I}^p\}.$$

So for all $\mathbf{g} \in \mathbf{K}_{I,x}$,
$\mathbf{g} = \sum_{i=1}^{p} \lambda_i \nabla_x j(x, \bar{\alpha}_i) = \sum_{i=1}^{p} \lambda_i \sum_{k=1}^{m} (\bar{\omega}_{i,k} - \omega_{i,k})\ \nabla_x j(x, \alpha_k)$. Then, $\mathbf{g}$ is a linear combination of vectors of $\mathbf{K}_{I_m,x}$, and consequently $\mathbf{g} \in \mathbf{K}_{I_m,x}$, so $\mathbf{K}_{I,x} \subset \mathbf{K}_{I_m,x}$. In addition, we already have $\mathbf{K}_{I_m,x} \subseteq \mathbf{K}_{I,x}$ by definition, and finally $\mathbf{K}_{I,x} = \mathbf{K}_{I_m,x}$. □

Remark 3.1 Theorem 3.1 shows that adding a new operating condition to a composite objective function does not necessarily provide any further information, but may only perturb its initial weights.

Remark 3.2 In a worst-case scenario, the dimension of $\mathbf{K}_{I,x}$ is n, which implies that the cardinality of $\mathbf{I_m}$ has to be at least n to ensure $\mathbf{K}_{I_m,x} = \mathbf{K}_{I,x}$. This depends on the properties of the functional $\nabla_x j$.

3.3 Induced Optimality Condition on Implicit Additional Conditions

The following theorem expresses a relationship between the first-order optimality, at a point x, of a weighted sum of m well-chosen operating conditions, and the functional variation of any derived multipoint problems built on $m+1$ conditions with specific weights in the neighborhood of x. It is a direct consequence of Theorem 3.1.

Theorem 3.2 *If $j \in C^1(\mathbb{R}^n \times \mathbf{I}, \mathbb{R})$, then Conditions 3.1 and 3.2 are equivalent.*

Proof Theorem 3.1 shares the Condition 3.1 with Theorem 3.2. The equivalence between Conditions 3.2 and 3.3 is proved here, which gives the equivalence between Conditions 3.1 and 3.3.

First, let us introduce two statements. For all weights $(\omega_1, \ldots, \omega_m) \in \mathbb{R}^m$ and operating conditions $(\alpha_1, \ldots, \alpha_m) \in \mathbf{I}^m$, let us suppose that we have $x \in \mathbb{R}^n$ such that

$$\sum_{k=1}^{m} \omega_k \ \nabla_x j(x, \alpha_k) = 0. \tag{5}$$

Next, for all $(\bar{\omega}_1, \ldots, \bar{\omega}_m) \in \mathbb{R}^m$, $\alpha_{m+1} \in \mathbf{I}$, $D \in \mathbb{R}^n$ and $t \in \mathbb{R}$, using a Taylor expansion, we have

$$\sum_{k=1}^{m+1} \bar{\omega}_k j(x + tD, \alpha_k) = \sum_{k=1}^{m+1} \bar{\omega}_k j(x, \alpha_k) + t\langle D, \sum_{k=1}^{m+1} \bar{\omega}_k \nabla_x j(x, \alpha_k)\rangle + \mathbf{O}(t^2), \tag{6}$$

where $\langle ., . \rangle$ is the scalar product.

Let us first prove that Condition 3.2 $\Rightarrow$ Condition 3.3. Thanks to the hypothesis of Condition 3.2, $\sum_{k=1}^{m+1} \omega_k \ \nabla_x j(x, \alpha_k) = \sum_{k=1}^{m} \bar{\omega}_k \ \nabla_x j(x, \alpha_k)$, and next, $\sum_{k=1}^{m} \omega_k \ \nabla_x j(x, \alpha_k) = \sum_{k=1}^{m} \bar{\omega}_k \ \nabla_x j(x, \alpha_k) - \omega_{m+1} \nabla_x j(x, \alpha_{m+1})$. Let us choose $\bar{\omega}_{m+1} = -\omega_{m+1}$, then $\sum_{k=1}^{m} \omega_k \ \nabla_x j(x, \alpha_k) = \sum_{k=1}^{m+1} \bar{\omega}_k \ \nabla_x j(x, \alpha_k)$. Using (5), for all $(\omega_1, \ldots, \omega_m) \in \mathbb{R}^m$, there exists $(\bar{\omega}_1, \ldots, \bar{\omega}_{m+1}) \in \mathbb{R}^m \times \mathbb{R}^*$ such that $\sum_{k=1}^{m+1} \bar{\omega}_k \ \nabla_x j(x, \alpha_k) = 0$. Replacing the previous equation in (6), we obtain $\sum_{k=1}^{m+1} \bar{\omega}_k \ j(x + t\ D, \alpha_k) = \sum_{k=1}^{m+1} \bar{\omega}_k \ j(x, \alpha_k) + \mathbf{O}(t^2)$.

Let us prove now that Condition 3.3 $\Rightarrow$ Condition 3.2. Assuming Condition 3.3, we have $\sum_{k=1}^{m} \omega_k \ \nabla_x j(x, \alpha_k) = 0$, and

$$\sum_{k=1}^{m+1} \bar{\omega}_k j(x + t\ D, \alpha_k) = \sum_{k=1}^{m+1} \bar{\omega}_k j(x, \alpha_k) + \mathbf{O}(t^2).$$

So, (6) becomes $0 = t\ \langle D, \sum_{k=1}^{m+1} \bar{\omega}_k \nabla_x j(x, \alpha_k)\rangle + \mathbf{O}(t^2)$. Due to the uniqueness of the Taylor expansion, every term of this t polynomial is null, then $0 = \ \langle D, \sum_{k=1}^{m+1} \bar{\omega}_k \ \nabla_x j(x, \alpha_k)\rangle$. This equality is true for all D in $\mathbb{R}^n$, so in particular for $D = \sum_{k=1}^{m+1} \bar{\omega}_k \ \nabla_x j(x, \alpha_k)$, which gives $\left\| \sum_{k=1}^{m+1} \bar{\omega}_k \ \nabla_x j(x, \alpha_k) \right\|^2 = 0$, and then $\sum_{k=1}^{m+1} \bar{\omega}_k \ \nabla_x j(x, \alpha_k) = 0$.

But, by hypothesis, $\sum_{k=1}^{m} \omega_k \nabla_x j(x, \alpha_k) = 0$, so finally $\sum_{k=1}^{m+1} \bar{\omega}_k \nabla_x j(x, \alpha_k) = \sum_{k=1}^{m} \omega_k \nabla_x j(x, \alpha_k)$, and this condition is equivalent to Condition 3.2. □

Interpretation In [34], the Pareto stationarity is defined as the existence of a convex combination of the gradient vectors that is null. When the objective is a smooth function of the design variables, it is a necessary condition for Pareto optimality. Theorem 3.2 shows that a Pareto stationary point, for a problem with the conditions $\{\alpha_k\}_{k\in\{1,\dots,m\}} \in \mathbf{I}^m$ such that $\mathbf{K}_{I_m,x} = \mathbf{K}_{I,x}$, is also Pareto stationary for any other problem with additional conditions taken from $\mathbf{I}$, as long as the weights of this new problem are positive. In other words, the theorem gives a necessary condition for the choice of the discrete set $\mathbf{I_m}$ for Pareto stationarity on the continuous set $\mathbf{I}$. If the weights are positive, it is also a sufficient condition.

On the sign of $\bar{\omega}_k$ In Theorems 3.1 and 3.2, it is not ensured that the weights $\bar{\omega}_k$ are positive. We have $\bar{\omega}_k = \omega_k + \rho_k$, and ρ_k is the component of the $m+1$th condition on the kth vector of the minimal gradient set. Then, a negative ρ_k means that the $m+1$th condition is concurrent with the kth one. When $\bar{\omega}_k$ is negative, increasing the objective on the kth condition will decrease the composite objective function. It means that the condition is locally concurrent with the objective, and its degradation leads to a better global performance in the sense defined by the initial problem.

To the authors' knowledge, this result is new and shows that (4) proposed in [3] is over restrictive. As a matter of fact, if the two vector spaces $\mathbf{K}_{I_m,x}$ and $\mathbf{K}_{I,x}$ are equal, then, when x is a local minimum of $J_m(x)$, there does not exist any perturbation of x that can improve the performance at first order, at any operating condition, without degrading it at least at another one. When the dimension of $\mathbf{K}_{I,x}$ is lower than the number of design variables, this means that incorporating more operating conditions than the number of design variables in the multipoint objective function is not a necessary condition for this property.

3.4 Second-Order Derivatives

The next theorem gives a more intuitive interpretation for Condition 3.1. It expresses that, if the derivative of the gradient vector with respect to α is inside the space $\mathbf{K}_{I_m,x}$, then this same space contains all gradients for α in $\mathbf{I}$. In addition, the two conditions are equivalent.We assume now that j is twice continuously differentiable.

Theorem 3.3 *If $j \in C^2(\mathbb{R}^n \times \mathbf{I}, \mathbb{R})$, then Conditions 3.1 and 3.4 are equivalent.*

Proof Let us first prove that Condition 3.1 ⇒ Condition 3.4. By definition of the derivative, for all $\alpha \in \mathbf{I}$ and $t \in \mathbb{R}$, such that $t + \alpha \in \mathbf{I}$, then $\lim_{t\to 0} \frac{\nabla_x j(x,\alpha+t) - \nabla_x j(x,\alpha)}{t} = \frac{\partial \nabla_x j(x,\alpha)}{\partial \alpha}$. We have $\nabla_x j(x, \alpha + t) \in \mathbf{K}_{I,x}$, and $\nabla_x j(x, \alpha) \in \mathbf{K}_{I,x}$. Also, $\mathbf{K}_{I,x}$ being a vector space, it contains any sum of two of its vectors, and $\frac{\nabla_x j(x,\alpha+t) - \nabla_x j(x,\alpha)}{t} \in \mathbf{K}_{I,x}$. Assuming Condition 3.1, we have $\mathbf{K}_{I_m,x} = \mathbf{K}_{I,x}$, so $\frac{\nabla_x j(x,\alpha+t) - \nabla_x j(x,\alpha)}{t} \in \mathbf{K}_{I_m,x}$.

The limit of a continuous function in a space being in the closure of this space, then $\lim_{t\to 0} \frac{\nabla_x j(x,\alpha+t) - \nabla_x j(x,\alpha)}{t} \in \overline{\mathbf{K}}_{I_m,x}$, and $\frac{\partial \nabla_x j(x,\alpha)}{\partial \alpha} \in \overline{\mathbf{K}}_{I_m,x}$. Because $\mathbf{K}_{I_m,x} \subseteq \mathbb{R}^n$, it is a finite dimension subspace of the normed vector space $\mathbb{R}^n$, thus $\mathbf{K}_{I_m,x}$ is closed, and $\overline{\mathbf{K}}_{I_m,x} = \mathbf{K}_{I_m,x}$. That finally gives the Condition 3.4.

Let us prove now that Condition 3.4 $\Rightarrow$ Condition 3.1. Since the derivatives of the cost functional j are continuous by hypothesis, for all $(\alpha, \alpha_0) \in \mathbf{I}^2$, $\int_{\alpha_0}^{\alpha} \frac{\partial \nabla_x j(x,t)}{\partial t} dt + \nabla_x j(x, \alpha_0) = \nabla_x j(x, \alpha)$. For all $k \in \mathbb{N}^*$, the Riemann sum S_k is defined as $S_k := \frac{\alpha - \alpha_0}{k} \sum_{i=1}^{k} \frac{\partial \nabla_x j}{\partial \alpha}(x, \alpha_0 + i \frac{\alpha - \alpha_0}{k})$, and, because $j \in \mathbf{C}^2(\mathbb{R}^n \times \mathbf{I}, \mathbb{R})$, $\lim_{k \to +\infty} S_k = \int_{\alpha_0}^{\alpha} \frac{\partial \nabla_x j(x,t)}{\partial t} \mathrm{d}t$.

Condition 3.4 expresses that for all α, $\frac{\partial \nabla_x j}{\partial \alpha} \in \mathbf{K}_{I_m,x}$. Because $\mathbf{K}_{I_m,x}$ is a vector space, we also have $S_k \in \mathbf{K}_{I_m,x}$. If we take now $\alpha_0 \in \mathbf{I_m}$, $\nabla_x j(x, \alpha_0)$ is in $\mathbf{K}_{I_m,x}$, and then $S_k + \nabla_x j(x, \alpha_0) \in \mathbf{K}_{I_m,x}$. The series $S_k + \nabla_x j(x, \alpha_0)$, with the same argument as in the first part of the proof, converges in the close finite dimension vector space $\mathbf{K}_{I_m,x}$, then

$$\lim_{k \to +\infty} S_k + \nabla_x j(x, \alpha_0) = \int_{\alpha_0}^{\alpha} \frac{\partial \nabla_x j(x,t)}{\partial t} \mathrm{d}t + \nabla_x j(x, \alpha_0) = \nabla_x j(x, \alpha) \in \mathbf{K}_{I_m,x}.$$

□

Theorem 3.3 shows that the variations of the function with respect to the operating conditions link the gradients at each operating condition. Exploiting this relation, when Condition 3.4 is ensured, gives the necessary and sufficient Condition 3.1 for Theorems 3.1 and 3.2. That shows how the parametric optimization problem is specific, compared to a multi-objective problem when the composite objectives come from different disciplines, with their own state equations, so when the multipoint problem does not come from one continuous function $j : (x, \alpha) \mapsto j(x, \alpha)$, but from m different $j_k : x \mapsto j_k(x)$ functions.

4 Gradient Span Analysis Algorithm

To use the previous theorems in practical applications, a minimal set of conditions $\mathbf{I_m}$ that verifies the condition $\mathbf{K}_{I_m,x} = \mathbf{K}_{I,x}$ for the multipoint optimization has to be built. Thus, the optimization problem will require a minimum amount of expensive computations of the functions $j(x, \alpha_k)$ and their gradients $\nabla_x j(x, \alpha_k)$. The gradient span analysis (GSA) method described in the algorithm 3 is proposed for building such a set. This algorithm is used to obtain an orthonormal basis of $\mathbf{K}_{I_m,x}$ such that $\mathbf{K}_{I_m,x}$ contains every vector of $\mathbf{K}_{I_M,x}$, where $\mathbf{I_M}$ is an initial sample of size M in $\mathbf{I}$, with a maximal relative error of ϵ, and m as small as possible.

The GSA algorithm is based on the repeated use of modified Gram-Schmidt (MGS) processes shown by the algorithm 2. The MGS process gives an orthonormal basis $\{q_j, j \in \{1, \ldots, M\}\}$ of a vector set, e.g., the gradient set $\{\nabla_x j(x, \alpha_k), \alpha_k \in \mathbf{I}_M\}$.

Algorithm 2: Modified Gram-Schmidt process Let M be a strictly positive integer and $\pi_u(v) := \frac{\langle v,u \rangle}{\langle u,u \rangle} u$ the projection operator on the singleton u.

Step 0. Set $j = 1$.

Step 1. If $j = M + 1$, stop; otherwise, set $q_j^1 := \nabla_x j(x, \alpha_j)$, and set $i = 1$.

Step 2. Update q_j^{i+1} via $q_j^{i+1} := q_j^i - \pi_{q_i}(q_j^i)$, and set $i = i + 1$.

If $i = j$, set $j = j + 1$, and if $\|q_j^{i+1}\| \neq 0$, set $q_j := \frac{q_j^{i+1}}{\|q_j^{i+1}\|}$

go to step 1.

The GSA algorithm relies on two rules which are to take into account more vectors than the dimension of the gradient space, and to find the minimal subset of the gradient set that spans $\mathbf{K}_{I,x}$ for a given projection error.The algorithm starts by building a fine sample $\mathbf{I_M}$ of the condition set $\mathbf{I}$ used to estimate $\mathbf{K}_{I,x}$. Then, conditions are chosen and added to the set $\mathbf{I_m}$ until $\mathbf{K}_{I_m,x} = \mathbf{K}_{I_M,x}$. At each step, the condition that maximizes the quantification of the intersection $\mathbf{K}_{I_M,x} \cap \mathbf{K}_{I_m,x}$ given by (7) is taken from available conditions in $\mathbf{I_M} \backslash \mathbf{I_m}$.

$$c(\mathbf{I_m}) = \text{card}\{\alpha \in \mathbf{I_M}, ||\nabla_x j(x,\alpha) - Proj(\nabla_x j(x,\alpha), \mathbf{K}_{I_m,x})|| < \epsilon ||\nabla_x j(x,\alpha)||\}. \tag{7}$$

A potential set $\mathbf{I_{m1}}$ will then be preferred to $\mathbf{I_{m2}}$ if $c(\mathbf{I_{m1}}) > c(\mathbf{Im2})$.

Two strategies are possible for the choice of the threshold ϵ. It can be either the stopping criteria $||\nabla J(x)|| < \epsilon$ of the optimization algorithm or the norm of the objective function gradient times a small constant, $\epsilon = 10^{-3}||\nabla_x J(x)||$.

Algorithm 3: Gradient Span Analysis Algorithm (GSA) Let $\epsilon \in \mathbb{R}$, $\epsilon > 0$, and $\mathbf{I_M} = \{\alpha_1, \ldots, \alpha_M\} \in \mathbf{I}^M$ be a set of operating conditions.

Step 0. Initialize the set of available operating conditions: set $\mathbf{A} = \mathbf{I_M}$.

Initialize the set of selected operating conditions: set $\mathbf{I_m} = \{\}$ and $j = 1$.

Step 1. Initialize the counter: set $\bar{c} = 0$. Initialize the set of tested operating conditions at the j-th iteration: $\mathbf{T_j} = \{\}$.

Step 2. Take a non-tested operating condition $\alpha \in \mathbf{A} \backslash \mathbf{T_j}$, and update the set of tested operating conditions $\mathbf{T_j} = \mathbf{T_j} \cup \{\alpha\}$. Set $q_j^1 = \nabla_x j(x,\alpha)$ and $i = 1$.

Step 4. Apply a MGS iteration on q_j^1:

while $i < j$, set $q_j^{i+1} = q_j^i - \pi_{q_i}(q_j^i)$ and $i = i + 1$.

Step 5. Set $c = 1$ and $p = 1$.

If $\|q_j^j\| = 0$, go to step 2; otherwise, set $q_j = \frac{q_j^j}{\|q_j^j\|}$.

Step 6. Set $v_{j,p}^0 = \nabla_x j(x,\alpha_p)$ and set $k = 0$.

Step 7. Project $\nabla_x j(x,\alpha_p)$ on span$\{q_1, \ldots, q_j\}$:

while $k < j$, set $v_{j,p}^{k+1} = v_{j,p}^k - \pi_{q_k}(v_{j,p}^k)$ and $k = k + 1$.

Step 8. If $\|v_{j,i}^j\| < \epsilon \|\nabla_x j(x,\alpha_p)\|$, set $c = c + 1$.

Step 9. If $p < M$, set $p = p + 1$, and go to step 6.

Step 10. If $c > \bar{c}$, set $\bar{c} = c$, $\bar{q} = q_j$, and $\bar{\alpha} = \alpha$.

Step 11. If $\mathbf{A} \backslash \mathbf{T_j} \neq \{\}$, go to Step 2; otherwise, set $q_j := \bar{q}$, and update the condition sets via $\mathbf{A} = \mathbf{A} \backslash \{\bar{\alpha}\}$ and $\mathbf{I_m} = \mathbf{I_m} \cup \{\bar{\alpha}\}$.

Step 12. If $c_m = M$, stop; otherwise, set $j = j + 1$, and go to Step 1.

Step 10. on the choice of the operating conditions can be replaced by

Step 10'. If $c = \bar{c}$ and $\min\{\|\alpha - \tilde{\alpha}\|, \tilde{\alpha} \in \mathbf{I_m}\} > \min\{\|\bar{\alpha} - \tilde{\alpha}\|, \tilde{\alpha} \in \mathbf{I_m}\}$, or $c > \bar{c}$, then set $\bar{c} = c$, $\bar{q} = q_j$, and $\bar{\alpha} = \alpha$. This gives the preference to operating conditions that are more distant to the already selected ones when they lead to identical gradient spans.

Minimality of the set $\mathbf{I_m}$ It is not proven that the set of operating conditions $\mathbf{I_m}$ given by the GSA algorithm is minimal. Nevertheless, it is guaranteed that the added condition is optimally chosen at each step. But at the end of the process, another set of lower dimension that also spans $\mathbf{K}_{I,x}$ with the same maximal error could exist. To ensure this global optimality, all combinations of vectors should be tested, which would lead to the orthonormalization of $\binom{M}{n}$ sets.This number quickly becomes too high to use the method. For instance, the analysis of a typical 3D wing optimization problem with 100 variables and 10 conditions would require 10^{13} Gram-Schmidt processes.

5 Computation of an Equivalent Problem

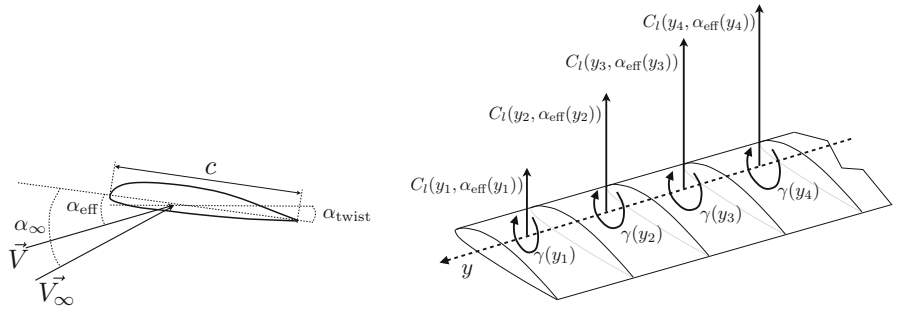
In this section, given an arbitrary multipoint problem with a set of M operating conditions, we aim at building the weights of an equivalent problem with a minimal set of m operating conditions selected by the Algorithm 3. Thanks to the GSA algorithm, the subset $\mathbf{I_m} := (\bar{\alpha}_1, \ldots, \bar{\alpha}_m)$ of $\mathbf{I_M}$ is built such that $\mathbf{K_{I_m,x}} = \mathbf{K_{I_M,x}}$. The gradient of the initial objective function is defined as $\nabla_x J(x) := \sum_{k=1}^{M} \omega_k \, \nabla_x j(x, \alpha_k)$. This quantity can be computed from less operating conditions, and with modified weights. For all $\alpha_k \in \mathbf{I}$ and $\omega_k \in \mathbb{R}$, there exists $(\omega_{k,1}, \ldots, \omega_{k,m}) \in \mathbb{R}^m$ such that $\omega_k \nabla_x j(x, \alpha_k) = \sum_{p=1}^{m} \omega_{k,p} \, \nabla_x j(x, \bar{\alpha}_p)$. A set of M similar linear systems is defined by (8). For all $k \in \{1, 2, \ldots, M\}$,

$$\omega_k \nabla_x j(x, \alpha_k) = \begin{bmatrix} \frac{\partial}{\partial x_1} j(x, \bar{\alpha}_1) & & \cdots & & \frac{\partial}{\partial x_1} j(x, \bar{\alpha}_m) \\ & \ddots & & & \\ & & \frac{\partial}{\partial x_i} j(x, \bar{\alpha}_p) & & \\ & & & \ddots & \\ \frac{\partial}{\partial x_n} j(x, \bar{\alpha}_1) & & \cdots & & \frac{\partial}{\partial x_n} j(x, \bar{\alpha}_m) \end{bmatrix} \begin{bmatrix} \omega_{k,1} \\ \cdots \\ \omega_{k,p} \\ \cdots \\ \omega_{k,m} \end{bmatrix}. \quad (8)$$

The matrix is not necessarily square ($m \leq n$), so a least-square solver is used. However, the GSA algorithm builds $\mathbf{K}_{I_m,x}$ such that $\mathbf{K}_{I_m,x} = \mathbf{K}_{I_M,x}$, then for all $\alpha_k \in \mathbf{I}$, $\nabla_x j(x, \alpha_k) \in \text{span}\{\nabla_x j(x, \alpha_1), \ldots, \nabla_x j(x, \alpha_m)\}$, and the problem has at least one solution. Solving the M linear systems of (8) gives finally $\nabla_x J(x) = \sum_{k=1}^{M} \sum_{p=1}^{m} \omega_{k,p} \nabla_x j(x, \bar{\alpha}_p)$. Because $\nabla_x j(x, \bar{\alpha}_p)$ does not depend on k, a new set of weights can be defined as $\bar{\omega}_p = \sum_{k=1}^{M} \omega_{k,p}$. The gradient of the objective function can be computed as follows, $\nabla_x J(x) = \sum_{p=1}^{m} (\sum_{k=1}^{M} \omega_{k,p}) \nabla_x j(x, \bar{\alpha}_p) = \sum_{p=1}^{m} \bar{\omega}_p \nabla_x j(x, \bar{\alpha}_p)$, and then $\nabla_x J(x) = \sum_{k=1}^{M} \omega_k \, \nabla_x j(x, \alpha_k)$. We have $m \leq M$, as a consequence the method saves $M - m$ costly evaluations of the gradient. It can be noted again that the case $dim(\mathbf{K}_{I_M,x}) = n$, which leads to $m = M$, is the worst one.

6 Application of the Concept

In this section, Theorems 3.1 and 3.2 are applied to a multipoint wing optimization problem. The objective is to minimize the drag of a wing over a range of lift coeffi-



(a) An airfoil model **(b)** The wing model

Fig. 2 The geometry and variables of the lifting line model

cients using a gradient-based algorithm. A lifting line model estimates the lift and the drag [35]. The design variables are the twist angles of airfoils, and the gradients are computed using the discrete adjoint method.

6.1 Continuous Lifting Line Model

The lifting line theory is a linearized fluid model that gives an estimation of wing performance. More precisely, it enables the computation of the lift and drag induced by the lift. The flow around the wing is assumed to be potential, so non-viscous and at low Mach number, and the rotational of the velocity is supposed to be null. Potential sources are associated with wing sections (i.e., airfoils). Their interferences model the interactions between airfoils. In the present model, the state variables are the circulations $\gamma(y) := \frac{1}{2} C_l(y, \alpha_{\text{eff}}(y))\, c(y)$, where $C_l(y, \alpha_{\text{eff}}(y))$ is the lift coefficient of the airfoil of chord $c(y)$ at span y and angle of attack $\alpha_{\text{eff}}(y)$. The effective angle of attack given by $\alpha_{\text{eff}} := \alpha_\infty + \alpha_{\text{twist}} - \alpha_{\text{ind}}$ is the difference between the geometrical angle of attack and the angle of attack induced by the lift as shown in Fig. 2a. The infinite angle of attack is the angle between the airfoil and the infinite air speed. The effective angle of attack is the angle between the local air speed and the airfoil. The geometrical angle of attack is the sum of the wing angle of attack and the twist that is a local rotation of the airfoil around y axis; see Fig. 2b.

Induced angles of attack, due to downwash velocities, are computed by summing the effects of all airfoil circulations. It is defined as $\alpha_{\text{ind}}(y_0) := \frac{1}{4\pi} \int_{-s}^{s} \frac{\mathrm{d}\gamma(y)}{\mathrm{d}y} \frac{\mathrm{d}y}{y - y_0}$, where s is the half wingspan and y_0 is the current airfoil position. The downwash velocity $\mathbf{V} - \mathbf{V}_\infty$ is the difference between the air speed of the aircraft and the local air speed of wing sections, as shown in Fig. 2a.The circulation on each airfoil must then be computed.

Let us define the residual function $R(\gamma, y_0) := \frac{1}{2} C_l \left(y_0, \alpha_\infty + x(y_0) - \frac{1}{4\pi} \int_{-s}^{s} \frac{\mathrm{d}\gamma(y)}{\mathrm{d}y} \frac{\mathrm{d}y}{y - y_0} \right) c(y_0) - \gamma(y_0)$, where the local twist α_{twist} is taken as the design variable $x(y_0)$. This residual function is the difference between a given circulation γ, and the circulation given by the lifts, under effective angles of attack due to that same circulation. To solve the integro-

differential equation $R(\gamma, y_0) = 0$, a Newton's method is used, taking the circulation as unknowns.We take here the twist angles as design variables because they drive the lift repartition for a given planform, i.e., fixed chords and airfoil positions. Finding the optimal twist vector is a typical lift repartition design problem in aerodynamics, and one of the main lifting line model application.

6.2 Discrete Model and its Computation

A cartesian discretization of y with a step Δy gives the following discrete vectors of size $p = 2s/\Delta y$, the twist angles x used as design variables, $x := \left[x_1, \ldots, x_i, \ldots, x_p\right]^T := \left[x(y_1), \ldots, x(y_i), \ldots, x(y_p)\right]^T$, the chords $c := \left[c_1, \ldots, c_i, \ldots, c_p\right]^T := \left[c(y_1), \ldots, c(y_i), \ldots, c(y_p)\right]^T$, and the circulations $\Gamma := \left[\gamma_1, \ldots, \gamma_i, \ldots, \gamma_p\right]^T := \left[\gamma(y_1), \ldots, \gamma(y_i), \ldots, \gamma(y_p)\right]^T$. The induced angles of attack are then $\alpha_{\text{ind},i} = \frac{1}{4\pi} \sum_{k=0,k\neq i}^{p} \frac{\mathrm{d}\gamma_k}{\mathrm{d}y} \frac{\Delta y}{y_k - y_i}$, and residuals become

$$R(\Gamma, x) = \begin{bmatrix} R_1 \\ \cdots \\ R_i \\ \cdots \\ R_p \end{bmatrix} = \frac{1}{2} \begin{bmatrix} C_{l1}(\alpha_\infty + x_1 - \frac{1}{4\pi} \sum_{k=0,k\neq 1}^{p} \frac{\mathrm{d}\gamma_k}{\mathrm{d}y} \frac{\Delta y}{y_k - y_1}) c_1 \\ \cdots \\ C_{li}(\alpha_\infty + x_i - \frac{1}{4\pi} \sum_{k=0,k\neq i}^{p} \frac{\mathrm{d}\gamma_k}{\mathrm{d}y} \frac{\Delta y}{y_k - y_i}) c_i \\ \cdots \\ C_{lp}(\alpha_\infty + x_p - \frac{1}{4\pi} \sum_{k=0,k\neq p}^{p} \frac{\mathrm{d}\gamma_k}{\mathrm{d}y} \frac{\Delta y}{y_k - y_p}) c_p \end{bmatrix} - \Gamma.$$

The Newton's method gives the next iterate, using the derivatives of the residual with respect to the state variables, as follows $\Gamma_{n+1} := \Gamma_n - (1 - \beta) \left[\frac{\partial R}{\partial \Gamma}(\Gamma, x)\right]^{-1} \Gamma$. The relaxation factor β is used to numerically stabilize the method.

6.3 Discrete Adjoint and Computation of Gradients

In the discrete adjoint formulation, the gradient of the residual with respect to the state variables is used in order to compute the gradient of the objective function, $J(\Gamma, x)$, with respect to the design variables x. Geometrical parameters, such as chords, twist angles, or airfoil positions, can be used as design parameters. The linear adjoint equation is given by $\frac{\partial J(\Gamma,x)}{\partial \Gamma} + \lambda_J^T \frac{\partial R(\Gamma,x)}{\partial \Gamma} = 0$, which solution gives the adjoint vector λ_J. Finally, the gradient is assembled as

$$\nabla_x J(\Gamma, x) = \frac{\partial J(\Gamma, x)}{\partial x} + \lambda_J^T \frac{\partial R(\Gamma, x)}{\partial x}. \tag{9}$$

The derivatives of the residuals with respect to the circulation are given by the following equations. For all $(i, j) \in \{1, \ldots, p\} \times \{1, \ldots, p\}$,

$$\frac{\partial R_i(\Gamma, x)}{\partial \gamma_j} = \frac{1}{2} \frac{\partial \alpha_{\text{ind},i}}{\partial \gamma_j} \frac{\mathrm{d}C_{li}}{\mathrm{d}\alpha} (\alpha_\infty + x_i - \frac{1}{4\pi} \sum_{k=0,k\neq i}^{p} \frac{\mathrm{d}\gamma_k}{\mathrm{d}y} \frac{\Delta y}{y_k - y_i}) c_i - \delta_{i,k}.$$

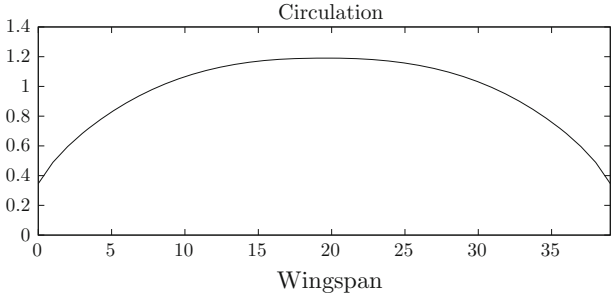


Fig. 3 Optimal circulation for a single-point wing design at C_l=0.5

The derivatives of the induced angle of attack with respect to the circulation are given by $\frac{\partial \alpha_{\text{ind},i}}{\partial \gamma_j} = \frac{1}{4\pi} \sum_{k=0,k\neq i}^{p} \frac{\mathrm{d}}{\mathrm{d}\gamma_j}(\frac{\mathrm{d}\gamma_k}{\mathrm{d}y}) \frac{c_i}{y_j - y_i}$, where the derivative $\frac{\mathrm{d}}{\mathrm{d}\gamma_j}(\frac{\mathrm{d}\gamma_k}{\mathrm{d}y})$ depends on the discretization scheme used for the computation of $\frac{\mathrm{d}\gamma_k}{\mathrm{d}y}$. Here, second-order centered finite differences are used. The derivatives of the residuals with respect to the design variables are computed with $\frac{\partial R_i(\Gamma,x)}{\partial x} = \frac{1}{2}\frac{\mathrm{d}C_{l_i}}{\mathrm{d}\alpha}(\alpha_\infty + x_i - \sum_{k=0,k\neq i}^{p} \frac{\mathrm{d}\gamma_k}{\mathrm{d}y} \frac{\Delta y}{y_k - y_i}) c_i$, where the functions of interest, the lift and the drag, are, respectively, given by

$$C_l = \frac{2\Delta y}{S} \sum_{i=0}^{p} \gamma_i, \tag{10}$$

and

$$Cd_{\text{ind}} = \frac{2\Delta y}{S} \sum_{i=0}^{p} \gamma_i \alpha_{\text{ind},i}. \tag{11}$$

The source terms of the adjoint equations for lift and drag are $\frac{\partial C_l}{\partial \gamma_p} = \frac{2\Delta y}{S}$, and $\frac{\partial Cd_{\text{ind}}}{\partial \gamma_p} = \frac{2\Delta y}{S}[\alpha_{\text{ind},p} + \sum_{i=0}^{p} \gamma_i \frac{\partial \alpha_{\text{ind},i}}{\partial \gamma_k}]$. The so-called geometrical terms required for the gradients computation $\frac{\partial J(\Gamma,x)}{\partial x}$ in (9) are null because the functions (10) and (11) have no direct dependance to the design variables.

6.4 GSA and Lifting Line Model

The fuel burn rate of an aircraft is proportional to the drag while the lift is imposed by the aircraft mass. From the lifting line model, a parametric objective function $Cd_{\text{ind}}(x, \alpha_\infty)$, a constraint $C_l(x, \alpha_\infty)$, and their gradients $\nabla_x Cd_{\text{ind}}(x, \alpha_\infty)$ and $\nabla_x C_l(x, \alpha_\infty)$ are obtained. It is possible to apply the GSA approach to this particular problem, with here 40 design variables. According to the lifting line theory, the circulation distribution at the optimum of a single-point problem is elliptical. This is observed in Fig. 3 for the problem $\min_{x\in\mathbb{R}^n} Cd_{\text{ind}}(x, \alpha_\infty)$ subject to $C_l(x, \alpha_\infty) = C_{l0}$.

With a null sideslip angle, the problem is symmetric, and only a half of the model may be considered. The whole model is kept in our example.

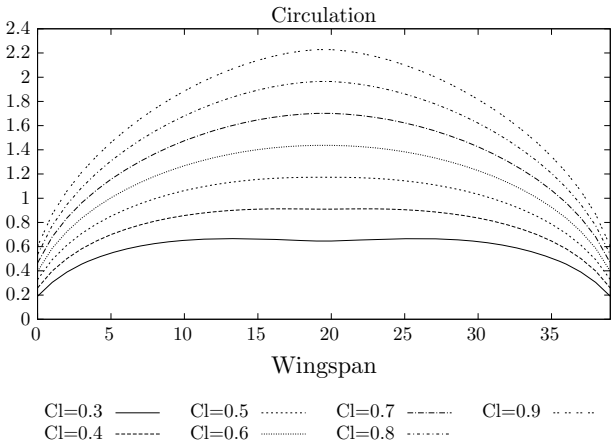


Fig. 4 Optimal circulation for a multipoint wing design from $C_l = 0.3$ to $C_l = 0.9$

Three numerical experiments are conducted with the lifting line model. In the first one, the GSA algorithm performs an analysis of the problem on a lift range of [0.3, 0.9]. Drag coefficients and their gradients on five hundred samples of the parameter α_∞ are computed. As a result, the gradient span is of dimension 2. Therefore, only 3 operating conditions are required to formulate the robust optimization problem. Because the design space is of dimension 40, this approach shows a gain of 38 computations compared to a composite objective function with a number of operating conditions under the hypothesis (4), which represents a cost cut of 93 %. Similar calculations are performed with 80 design variables. The gradient span dimension for the drag is also 2. In a general case, when the gradient span dimension is close or equal to the number of design variables, increasing the number of design variables can increase the gradient span dimension.

In the second experiment, a multipoint design problem is built with 7 operating conditions and a uniform weighting. The operating conditions are defined by lift constraints: $C_l = 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9$, and the angle of attack at each condition is adjusted by the optimization algorithm to ensure this minimal lift. This optimization problem is solved with the sequential least-square quadratic programming (SLSQP) algorithm [36]. The results are summarized in Fig. 4 for the circulation repartitions, and in Fig. 5 for the optimal twist vector x. 7 optimization iterations are required, which means 49 evaluations of the lifting line model.

In the last experiment, GSA is used to analyze the gradient span for each iteration of the second experiment, and the results are shown in Table 1. Finally, the equivalent problem given by the weights at convergence of the previous experiment is solved. This requires 7 optimization iterations, so 21 calls to the wing model. The difference between the circulations Γ of the optimum, given by the initial and equivalent problems, is of the same order of magnitude as the machine precision (10^{-14}) for the three conditions 1, 2, and 7. In other words, the optimizer has found the same physical solution to the two problems.

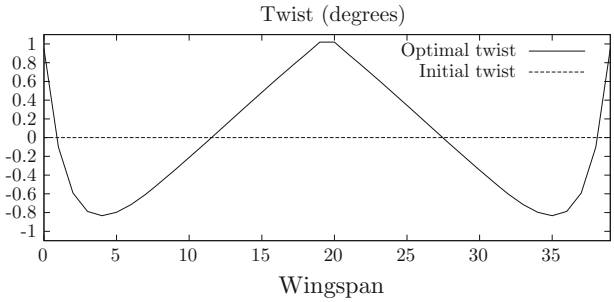


Fig. 5 Optimal twist repartition for a multipoint wing design from $C_l = 0.3$ to $C_l = 0.9$

Table 1 Original and equivalent multi conditions lifting line optimization problem

Iteration	ω_1	ω_2	ω_3	ω_4	ω_5	ω_6	ω_7
Baseline	1	1	1	1	1	1	1
Optimum	0.7509	3.2988	0	0	0	0	2.9501

By comparison of the two last experiments, the computational gain brought by the GSA approach is of 57 %. The comparison is valid as long as the computational cost of the equivalent weights is negligible compared to the model computational cost. This is true for high fidelity models and also for our low fidelity lifting line model.

On the other hand, the equivalent weight computation must be performed at the optimum because the theorems and GSA are applied at a given x. As a consequence, this requires the resolution of the initial problem, so this last experiment has only a theoretical and demonstrative interest.

However, it is important to notice that the dimension of the gradient span does not vary during the optimization. And even more interesting, the three conditions given by the GSA do not change during the optimization process, except at the first iteration, even if the equivalent weights are not constant. As a consequence, the analysis of the problem does not need to be performed for each iteration of the optimization process, but only after several ones, preserving the computational cost reduction. For the present problem, the operating conditions generating the gradient span are invariant during the optimization process.

The first step consists in performing a fine sampling of the operating condition ranges as in experiment 1. Latin hypercube sampling [37] or other design analysis of computer experiment [38] methods can be used for an efficient initial sampling. In the lifting line case, a uniform sampling is used. The initial sampling is followed by a GSA calculation that selects the required operating conditions to be included in the optimization problem. After that, a method for adequately choosing the weights is used. As a reminder, Normal Constraint [12], Successive Pareto Optimization [13], Directed Search Domain [14], or Multiple-Gradient Descent Algorithm [15] are possible approaches. Weights usually depend on the optimization iteration, so they are computed after the choice of the operating conditions. At the end of the optimization or during it, the gradient span dimension should be checked to be sure that Condition 3.1 is still valid.

7 Aircraft Wing Shape Optimization

The method described in this paper is applied here to a multipoint aircraft aerodynamic optimization using RANS simulations. This section is a summary of a design study that can be found in [23], while the present document focuses on mathematical demonstrations of the approach. The illustration shows that the concept is applicable to complex engineering systems with a hundred millions of degrees of freedom, and non-linear partial differential state equations. The XRF-1 glider model from Airbus is optimized in a space of operating conditions including multiple Mach numbers and multiple angles of attack. The 7 non-linear partial differential RANS equations that describe the flow physics are solved to estimate the performance of the configuration on a mesh with 20 millions of elements. A discrete adjoint strategy is used to efficiently compute its gradient with respect to the 90 design variables, in a way that is very similar to the lifting line model in Sect. 6. A computer aided design (CAD) model parametrizes the wing shape with engineering variables such as twist angles, thicknesses or wing section positions. The computational domain mesh and the geometry displayed in Fig. 6 give an idea of the case complexity. The GSA algorithm is used to select the operating conditions to be aggregated in an objective function, from the original operating conditions space; see Fig. 7. It is noteworthy that 6 operating conditions are

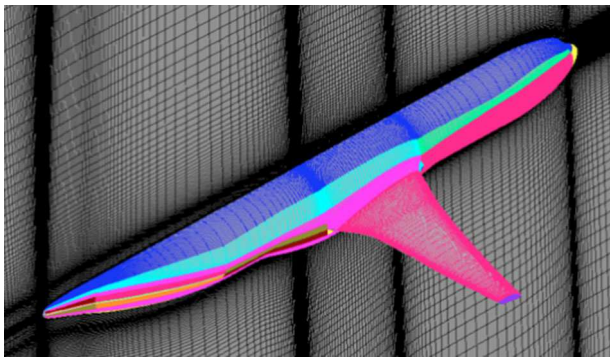


Fig. 6 Computational domain mesh of the XRF-1 model

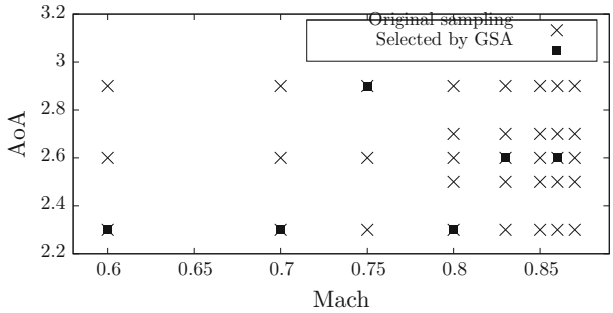


Fig. 7 Flight conditions selected for the XRF-1 multipoint optimization

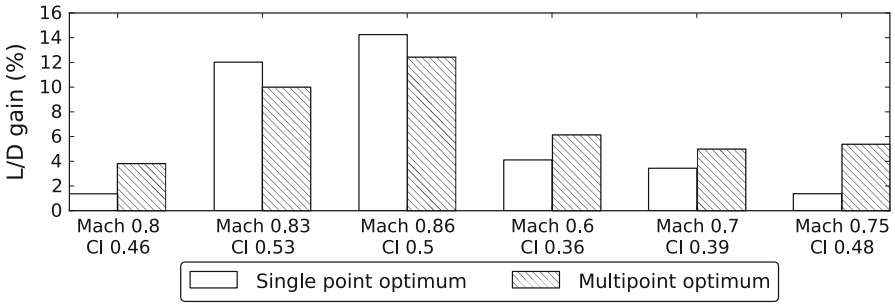


Fig. 8 XRF-1 single-point and multipoint optimization Cl/Cd gains compared to the baseline design

required for this non-linear problem with 90 design variables. The limited-memory Broyden–Fletcher–Goldfarb–Shanno algorithm handling bound constraints, L-BFGS-B [39], successfully performed the optimization. Lift-to-drag coefficients, defined by the ratio between lift and drag coefficients, are a major aerodynamic performance measure of an aircraft since fuel burn rate is inversely proportional to it. The single-point and multipoint optimization lift-to-drag performance gains, relative to the original design, are compared in Fig. 8. It shows that the multipoint optimization provides a compromise in performance, as opposed to the single operating condition optimization (Mach 0.83 and Cl 0.53).

The computational cost is 80,000 CPU hours on 400 Intel(R) Xeon(R) X5670 @ 2.93GHz processors, accounting for the initial sampling cost for the GSA algorithm, and the optimization run cost itself. This represents a 90% cost cut compared to an approach where "number of design variables +1" samples are selected in the operating condition ranges as in (4).

8 Conclusions

This paper demonstrates that the operating conditions are not just an input for a parametric optimization, but that the choice of these operating conditions is a part of the optimization problem formulation. The operating conditions incorporated in a weighted sum of objectives must be selected after the analysis of the objectives gradients. Ranges of operating conditions being the inputs, the choice of operating conditions is a sampling problem. The GSA algorithm is proposed to determine the required conditions and their number in order to minimize the computational cost of the optimization.

To summarize the main results, the multipoint optimization problem must combine m operating conditions, chosen such that the two gradient spanned vector spaces $\mathbf{K}_{I,x}$ and $\mathbf{K}_{I_m,x}$ are equal, otherwise one of the two following situations appears. If $m < dim(\mathbf{K}_{I,x})$, then, at the optimum, there exists at least one operating condition at which the objective can be improved without degrading the others, so the performance of the system can be enhanced. If $m > dim(\mathbf{K}_{I,x})$, then there exists an equivalent problem with less operating conditions, and modified weights, that gives the same solution, so computational resources are wasted. A counter-intuitive outcome is that

when the operating ranges are adequately sampled, i.e., $\mathbf{K}_{I,x} = \mathbf{K}_{I_m,x}$, modifying the operating conditions used in the optimization problem is equivalent to changing the weights of the composite objective function.

The methodology is validated on wing optimization, represented by a lifting line model, and using a discrete adjoint formulation. Three numerical experiments have been successfully conducted. They show the feasibility of the method, and its interest in terms of computational time savings, compared to state-of-the-art approaches. An example of modern transport aircraft wing optimization, based on high fidelity RANS simulations, confirms the range of applications.

Future work will address the impacts of using noisy gradients of the objective function, both on the GSA algorithm and the robust optimization results. This situation classically occurs in real-life optimization problems when adjoint calculations are not fully converged, or programs are not fully linearized, and in the case of finite precision arithmetics. Besides, the limits of the hypothesis stating that the operating conditions generating the gradient span are invariant during the optimization process will be tested. When the hypothesis is false, an iterative process made of an optimization followed by a GSA analysis can be used. However, the convergence of this algorithm is an open question. Finally, the weighted sum method being known to have limitations [27], the use of alternatives to solve the multi-objective optimization problem deserves further research. In particular, Theorems 1 and 2 suppose that a weighted function is being minimized; if not, generalizations would have to be demonstrated.

Acknowledgments The authors would like to thank Xavier Pinel for his mathematical feedbacks. Joël Brézillon and the reviewers are greatly acknowledged for their valuable remarks. This work was founded by Airbus and the Association Nationale de la Recherche et de la Technologie.

References

1. Hicks, R.M., Vanderplaats, G.N.: Application of numerical optimization to the design of supercritical airfoils without drag creep. In: Business Aircraft Meeting, Paper 770440. Society of Automotive Engineers, Wichita (1977)
2. Drela, M.: Pros and cons of airfoil optimization. In: Frontiers of Computational Fluid Dynamics, pp. 363–381. World Scientific, Singapore (1998)
3. Li, W., Huyse, L., Padula, S.: Robust airfoil optimization to achieve consistent drag reduction over a Mach range. Struct. Multidiscip. Optim. **24**(1), 38–50 (2002)
4. Beyer, H.G., Sendhoff, B.: Robust optimization—a comprehensive survey. Comput. Methods Appl. Mech. Eng. **196**(33–34), 3190–3218 (2007)
5. Campbell, R.L.: Efficient viscous design of realistic aircraft configurations. In: 29th AIAA, Fluid Dynamics Conference, 98–2539. American Institute of Aeronautics and Astronautics, Albuquerque (1998)
6. Reuther, J.J., Jameson, A., Alonso, J.J., Rimlinger, M.J., Saunders, D.: Constrained multipoint aerodynamic shape optimization using an adjoint formulation and parallel computers, part 1. J. Aircr. **36**(1), 51–60 (1999)
7. Reuther, J.J., Jameson, A., Alonso, J.J., Rimlinger, M.J., Saunders, D.: Constrained multipoint aerodynamic shape optimization using an adjoint formulation and parallel computers, part 2. J. Aircr. **36**(1), 61–74 (1999)
8. Buckley, H.P., Zhou, B.Y., Zingg, D.W.: Airfoil optimization using practical aerodynamic design requirements. J. Aircr. **47**(5), 1707–1719 (2010)

9. Jameson, A., Leoviriyakit, K., Shankaran, S.: Multi-point aero-structural optimization of wings including planform variations. In: 45th Aerospace Sciences Meeting and Exhibit, AIAA Paper 2007–764, Reno, Nevada (2007)
10. Zingg, D., Elias, S.: Aerodynamic optimization under a range of operating conditions. AIAA J. **44**(11), 2787–2792 (2006)
11. Das, I., Dennis, J.E.: Normal-boundary intersection: a new method for generating the Pareto surface in nonlinear multicriteria optimization problems. SIAM J. Optim. **8**(3), 631–657 (1998)
12. Messac, A., Ismail-Yahaya, A., Mattson, C.: The normalized normal constraint method for generating the Pareto frontier. Struct. Multidiscip. Optim. **25**, 86–98 (2003)
13. Mueller-Gritschneder, D., Graeb, H.E., Schlichtmann, U.: A successive approach to compute the bounded Pareto front of practical multiobjective optimization problems. SIAM J. Optim. **20**(2), 915–934 (2009)
14. Erfani, T., Utyuzhnikov, S.V.: Directed search domain: a method for even generation of the Pareto frontier in multiobjective optimization. Eng. Optim. **43**(5), 467–484 (2011)
15. Désidéri, J.A.: Mgda variants for multi-objective optimization. Tech. Rep. 8068, Rapport de Recherche INRIA (2012)
16. Hansen, N., Ostermeier, A.: Completely derandomized self-adaptation in evolution strategies. Evol. Comput. **9**(2), 159–195 (2001)
17. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002)
18. Marco, N., Désidéri, J.A., Lanteri, S.: Multi-objective optimization in CFD by genetic algorithms. Technical Report RR-3686, INRIA (1999)
19. Vicini, A., Quagliarella, D.: Inverse and direct airfoil design using a multiobjective genetic algorithm. AIAA J. **35**(9), 1499–1505 (1997)
20. Tang, Z., Désidéri, J.A., Périaux, J.: Multicriterion aerodynamic shape design optimization and inverse problems using control theory and Nash games. J. Optim. Theory Appl. **135**(3), 599–622 (2007)
21. Zingg, D.W., Nemec, M., Pulliam, T.H.: A comparative evaluation of genetic and gradient-based algorithms applied to aerodynamic optimization. Eur. J. Comput. Mech. Rev. Eur. Méc. Numér. **17**(1–2), 103–126 (2008)
22. Kenway, G.K.W., Martins, J.R.R.A.: Multipoint high-fidelity aerostructural optimization of a transport aircraft configuration. J. Aircr. **51**(1), 144–160 (2014)
23. Gallard, F., Meaux, M., Montagnac, M., Mohammadi, B.: Aerodynamic aircraft design for mission performance by multipoint optimization. In: 21st AIAA Computational Fluid Dynamics Conference, San Diego, USA (2013)
24. Marler, R.T., Arora, J.S.: Survey of multi-objective optimization methods for engineering. Struct. Multidiscip. Optim. **26**(6), 369–395 (2004)
25. Giannessi, F., Mastroeni, G., Pellegrini, L.: On the theory of vector optimization and variational inequalities. image space analysis and separation. In: Giannessi, F. (ed.) Vector Variational Inequalities and Vector Equilibria. Nonconvex Optimization and its Applications, vol. 38, pp. 153–215. Kluwer, Dordrecht (2000)
26. Zadeh, L.: Optimality and non-scalar-valued performance criteria. IEEE Trans. Autom. Control **8**(1), 59–60 (1963)
27. Das, I., Dennis, J.E.: A closer look at drawbacks of minimizing weighted sums of objectives for Pareto set generation in multicriteria optimization problems. Struct. Multidiscip. Optim. **14**, 63–69 (1997)
28. Lyu, Z., Kenway, G.K., Martins, J.: RANS-based aerodynamic shape optimization investigations of the common research model wing. In: AIAA Science and Technology Forum and Exposition (SciTech), National Harbor, MD (2014)
29. Carrier, G.: Single and multi-point aerodynamic optimizations of a supersonic transport aircraft wing using optimization strategies involving adjoint method and genetic algorithm. In: Proceedings of ERCOFTAC Workshop "Design optimization: methods and applications", Las Palmas (2006)
30. Kosambi, D.: Statistics in function space. JIMS **7**, 76–88 (1943)
31. My-Ha, D., Lim, K., Khoo, B., Willcox, K.: Real-time optimization using proper orthogonal decomposition: free surface shape prediction due to underwater bubble dynamics. Comput. Fluids **36**(3), 499–512 (2007)
32. Thanh, B.T., Damodaran, M., Willcox, K.: Proper orthogonal decomposition extensions for parametric applications in transonic aerodynamics (AIAA Paper 2003–4213). In: Proceedings of the 15th AIAA Computational Fluid Dynamics Conference (2003)

33. Braconnier, T., Ferrier, M., Jouhaud, J.C., Montagnac, M., Sagaut, P.: Towards an adaptive POD/SVD surrogate model for aeronautic design. Comput. Fluids **40**, 195–209 (2011)
34. Désidéri, J.A.: Multiple-gradient descent algorithm (MGDA) for multiobjective optimization. C. R. Math. **350**(56), 313–318 (2012)
35. Prandtl, L.: Theory of lifting surfaces. Tech. rep, NACA (1920)
36. Kraft, D.D.: A software package for sequential quadratic programming. Tech. Rep. DFVLR-FB 88–28, DLR German Aerospace Center, Institute for Flight Mechanics, Koln, Germany (1988)
37. McKay, M.D., Beckman, R.J., Conover, W.J.: Comparison of three methods for selecting values of input variables in the analysis of output from a computer code. Technometrics **21**(2), 239–245 (1979)
38. Sacks, J., Welch, W.J., Mitchell, T.J., Wynn, H.P.: Design and analysis of computer experiments. Stat. Sci. **4**(4), 409–423 (1989)
39. Zhu, C., Byrd, R.H., Lu, P., Nocedal, J.: Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. ACM Trans. Math. Softw. **23**(4), 550–560 (1997)