Dynamic Geometry Control for Robust Aerodynamic Shape Optimization

Gregg M. Streuber * and David. W. Zingg †
Institute for Aerospace Studies, University of Toronto, Toronto, Ontario, M3H 5T6, Canada

This work presents novel progressive and adaptive dynamic geometry control algorithms which seek to improve convergence and reduce user workload by partially automating the design of effective geometry control systems for aerodynamic shape optimization. These algorithms function by beginning in a coarse design space and periodically refining the geometry control with additional design variables when objective improvement becomes asymptotic. When refinement is initiated, progressive geometry control moves through a pre-defined sequence of increasingly fine geometry control schemes, while the adaptive algorithm instead dynamically generates a refined search space. This is accomplished by generating a list of candidate refinements and ranking them based on the minimum of a constrained quadratic suboptimization problem which constitutes an estimate of the maximum objective reduction possible in each candidate search space. The accuracy of this method is first validated on two inviscid problems, after which the progressive and adaptive algorithms are applied to two common aerodynamic shape optimization problems based on the Reynolds-Averaged Navier-Stokes equations, the twist and section optimization of the common research model wing-only geometry, and the planform optimization of a hybrid wing-body aircraft. In both cases, the dynamic geometry control schemes are able to converge to lower drag, often with fewer optimization iterations, compared to the tested static schemes.

I. Introduction

Aerodynamic shape optimization is a field coupling numerical optimization with computational fluid dynamics (CFD), where the output of a CFD simulation around an aerodynamic body is optimized with respect to the shape of the body. With the adoption of the discrete adjoint method [1], gradients can be computed at a cost almost independent of the number of design variables, making gradient-based optimization a fast, cost-effective solution. Combined with the rise of affordable computing, this has made aerodynamic shape optimization practical for the optimization of full aircraft designs in high fidelity and has led to its widespread adoption in industry and academia for the analysis and improvement of aircraft [2] in a variety of settings including airfoils and wings [3–7], full conventional configurations [8–10] and unconventional designs such as Hybrid Wing-Bodies (HWB) [11, 12], lifting-fuselages [13], strut-braced wings [14, 15] and box wings [16].

The rise of aerodynamic shape optimization is coincident with, and in part driven by, mounting pressure on commercial aerospace to produce a step reduction in emissions [17]. Efforts to produce such a reduction can be broadly categorized into three main streams: lighter materials, low-emissions propulsion, and improved conventional or unconventional aerodynamics. The benefits of lighter, stronger composite materials have been demonstrated in recent designs like the Boeing 787 [18] or Airbus A220 [19], while efforts on the propulsive side include higher bypass-ratio turbofans [20] or alternative fuels [21]. Directly reducing the drag of conventional aircraft through improved aerodynamics has been assisted by the application of aerodynamic shape optimization to detailed design. Such optimization is performed later in the design process when a preliminary design has already been developed and further geometric changes are limited.

However, without new technologies these methods are not sufficient to produce the gains desired by industry and regulators. Conventional aerodynamics in particular are rapidly approaching a point of diminishing returns as 70 years of optimizing tube-and-wing aircraft leaves little room for dramatic improvements. This has spurred interest in unconventional aircraft offering the potential for much larger improvements in performance, but discovering and analyzing them requires the application of aerodynamic shape optimization in the preliminary or exploratory stage prior

*Ph.D. Candidate, gregg.streuber@utoronto.ca
†University of Toronto Distinguished Professor of Computational Aerodynamics and Sustainable Aviation, Director, Centre for Research in Sustainable Aviation, and Associate Fellow AIAA, dwz@oddjob.utias.utoronto.ca

to detailed design. Unconventional, exploratory problems are by necessity characterized by large geometric flexibility combined with novel, unknown design spaces; these problems can be expensive and poorly-conditioned, making them difficult for the optimizer to navigate.

At the same time as this push for unconventional aircraft, the CFD models underpinning aerodynamic shape optimization codes are growing in cost and complexity, incorporating advanced tools like transition prediction [22] or moving from primarily inviscid optimization [23] to the current predominance of optimization based on the Reynolds-Averaged Navier-Stokes (RANS) equations [5, 15, 24], some authors have even taken the first steps towards enabling large eddy simulation for optimization [25]. Such improvements are necessary, but increase the cost of flow evaluation. The field then is at a nexus where design spaces are increasingly difficult and novel, and the costs of slow convergence are increasingly large. Clearly, there is a benefit to reducing both upfront costs, in terms of the time and experience requirements to formulate an effective aerodynamic shape optimization problem, and the ongoing costs of converging the problems themselves. We refer to this as improving robustness, which we define in this work as "the ability of an algorithm to reliably and consistently locate globally optimal designs with minimal user intervention". Previous work [26] has begun addressing the globally optimal component of this definition, while this work instead focuses on the idea of minimizing user intervention and improving efficiency through the use of dynamic geometry control.

In aerodynamic shape optimization, geometry control is the system by which the optimizer manipulates and deforms the aerodynamic surface which is being optimized. CAD-based approaches have been used previously [27], but in recent years many authors have adopted CAD-free geometry control due its superior flexibility and customizability. CAD-free geometry control in the literature encompasses many different methods, including traditional [28] and axial-augmented [14] Free-Form Deformation (FFD), B-spline patches [29], Hicks-Henne bump functions [30], component-based control [31], radial-basis functions [32], and numerous other specialized geometry control methods tailored to the requirements of their users [33–36]. Even within each category there exists significant variation between methods, including hybrid methods or even attempts to leverage dimensionality reduction [37] or machine learning [38] to find the most efficient geometry control schemes. While the details of these approaches differ, they are similar in that they are overwhelmingly applied in a static fashion. That is, the geometry control, whatever its specifics, is designed by a user prior to optimization and while design variables can change during optimization the topology of the control scheme remains fixed. Such static systems are simple to implement and have been successfully utilized in a vast array of different settings, but they do have some inherent drawbacks.

First, it can be observed that in many optimization problems the optimizer tends to begin with coarse, large-scale deformations and then steadily moves to smaller and smaller shape refinements as optimization proceeds. Having too few design variables will stunt objective improvement, while starting with too many can lead to a poorly conditioned optimization problem and poor convergence [39]. An example of this is shown in Figure 1, which plots the drag convergence for five variants of the same case: the inviscid optimization of a three-dimensional wing in transonic flow. Each variant differs only in the number of design variables available to the optimizer, and each was run until convergence or for a maximum of 300 design iterations.

As we would expect, the coarsest version (with just 51 design variables) converges quickly and smoothly but to a relatively high drag. Increasing the number of design variables to 163 produces clear improvements to final drag at minimal cost; another increase to 546 produces much smaller reductions in final drag and a slight slowdown in convergence. In the final two versions, however, the progressively larger number of design variables begin to significantly impede performance, with much higher drag after 300 iterations than the coarser 163 and 546 DV cases. If allowed to run to convergence, these cases would likely converge to superior optima than the coarser cases but this would take a prohibitively long time. There is also no guarantee that the optimizer would not stall or otherwise fail before convergence is achieved. Depending on the accuracy of the flow solver and the time or resources available there will be an ideal static geometry control scheme for a given problem; however, it will rarely be known a priori.

A second potential issue with static control, particularly in the context of preliminary or exploratory design, is that it can be overly restrictive. Any geometry control scheme can be thought of as a second set of constraints on optimization: an optimizer can only manipulate the design variables it is given, and fitting a static geometry control system over a continuous problem inherently invalidates certain search directions and biases the optimizer towards others. While exploratory optimization certainly can and has been performed with static geometry control, one must very carefully balance the contradictory goals of exploration and speed.

Dynamic geometry control (DGC) is an alternative in which the geometry control topology is modified throughout optimization, usually through refinement. Such a system is more complicated to implement, but can sidestep these issues by starting the optimization in a coarse search space and refining periodically throughout. This allows the

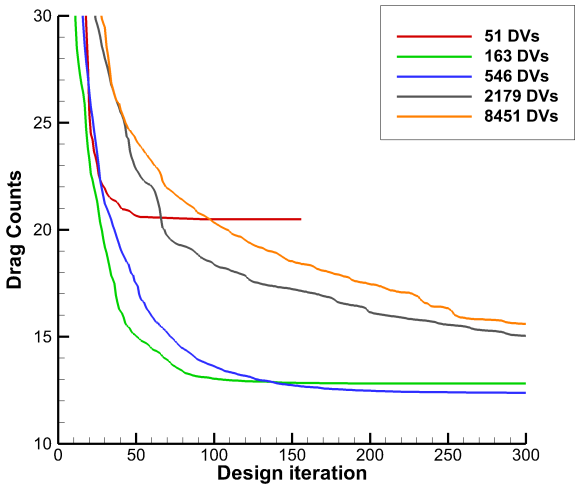


Fig. 1 Effect of number of design variables (DVs) on optimizer performance

optimization to maximize initial speed with a coarse geometry control, while adding finer control later to exploit the design space. Additionally, by leveraging sensitivity information from a gradient-based optimization it is possible to guide the refinement process and focus new control in regions of greatest potential, increasing the flexibility of the optimizer and preventing overly-constrictive static schemes.

Two particular flavours of DGC are considered in this work: progressive and adaptive control. Both are similar in that an initially coarse geometry control scheme is systematically refined throughout optimization, but take different approaches to this general idea. Progressive geometry control is relatively straightforward: starting with a coarse design space the optimizer steadily moves through a predefined sequence of progressively finer geometry control schemes. Some early work in this field [40, 41] examined both uniform refinement schemes and multigrid-like v-cycles to improve convergence while others such as Andreoli et al. [42] solely consider monotonic refinement. These methods have been successfully applied in stochastic optimization [43] but they have also been shown to work effectively in gradient-based optimization [44–46] where the costs of increased dimensionality are less explicit. In general, a common trait among progressive schemes is that the geometry control systems are designed beforehand by a user, or generated at runtime according to a fixed pattern.

Adaptive control, on the other hand, leaves the design of the new geometry control system largely up to automatic processes. Duvigneau et al. [43] developed an adaptive scheme built on the idea that a Bezier curve parameterization with a more regularized control polygon will be more effective than a less-regular alternative. A more advanced approach in the context of gradient-based optimization is to leverage readily available gradient data to optimize the expected performance of the new geometry control scheme. This approach was examined by Han and Zingg [47], who assessed candidate control schemes based on the value of the objective gradient, a higher gradient suggesting a greater potential for objective improvement. Masters et al. [48] produce good results using a similar method for 2-D airfoil optimization, in which the candidates are ranked based on a projection of the objective gradient onto the aerodynamic surface. He et al. [49] built further on the gradient-ranking concept in 2-D by using constraint gradients to penalize the objective gradient in the ranking. The current state of the art in the field, however, is represented by the work of Anderson and colleagues [44, 50, 51], who performed a quadratic fit of the design space for each candidate and took the constrained minimizer of this quadratic as an estimate of its maximum objective improvement, representing the first apparent attempt in the literature to directly approximate the shape of a candidate design space. This approach was adapted by Sinsay and Alonso [52] who used a genetic algorithm to determine the optimal refinement strategy.

This work furthers the investigation and application of these methods by implementing them in the context of a two-level axial and FFD geometry control system, building on the quadratic fit approach of Anderson and colleagues with improvements to the treatment of constraints and a novel approach to the approximation of candidate Hessians. Additionally, a significant effort is made to study and quantify the benefits and applicability of DGC through application of the developed progressive and adaptive algorithms to a variety of problems ranging from simple model problems to

full-scale cases representative of detailed or preliminary design.

II. Jetstream Optimization Framework

This study is undertaken using the Jetstream optimization framework, which is described below, with particular attention given to the FFD geometry control system.

A. Optimization Overview

Flow solutions in Jetstream are obtained through Diablo, a three-dimensional structured multiblock CFD solver capable of solving both the Euler [23] and RANS [53] equations; for the RANS equations, the Spalart-Allmaras one-equation turbulence model is used. Second-order summation-by-parts operators and scalar or matrix numerical dissipation are used for spatial discretization with simultaneous approximation terms weakly enforcing boundary conditions and block interfaces. The steady-state solution is obtained with a parallel Newton-Krylov-Schur algorithm utilizing an approximate-Newton phase for the initial iterate of a subsequent inexact-Newton phase. Linear systems are solved using the Generalized Minimum Residual (GMRES) method.

Gradients are obtained using an implementation [54, 55] of the discrete adjoint method [1]. A flexible variant [54] of the GCROT (generalized conjugate residual with inner orthogonalization and outer truncation) Krylov method [56] is leveraged to obtain adjoint solutions. These gradients are used to obtain subsequent iterates using the sequential-quadratic-programming algorithm SNOpt [57], with Hessians obtained via the Broyden–Fletcher–Goldfarb–Shanno (BFGS) [58–61] update method.

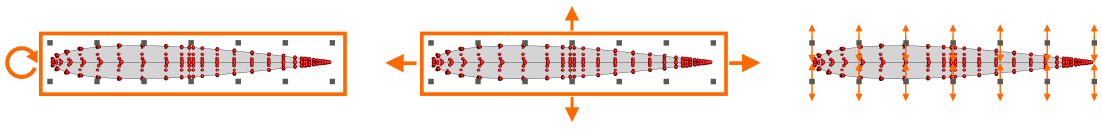
A linear elasticity method adapted by Hicken and Zingg [54] from the work of Truong et al. [62] handles mesh movement. Computational performance is improved by fitting the computational mesh with a coarse B-spline control mesh, the mesh deformation being applied to this control mesh, which is then used to generate a deformed computational mesh.

B. Static Free Form Deformation

Free-Form Deformation (FFD) [63], is a geometry control scheme which introduces an additional layer of abstraction between the surface and the design variables and enables consistent, intuitive design variables to be used for almost any type of surface geometry. FFD works in principle by embedding the surface to be deformed within an FFD volume; the optimizer then manipulates the FFD volume without needing to have any knowledge of what lies within it. As the FFD volume is deformed, the underlying geometry is deformed in turn - akin to manipulating a rubber block with a shape embedded inside. While it first gained traction with the animation community, FFD's use in aerodynamic shape optimization dates to 2004 [64].

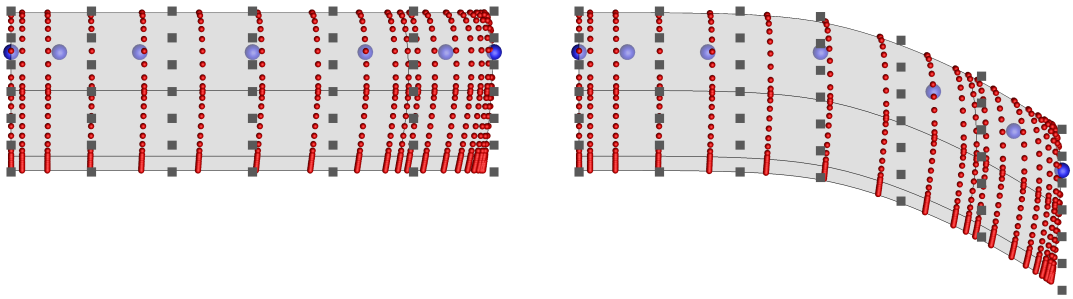
In this work, the embedded geometry is parameterized with B-spline surface patch control points and the FFD volumes they are embedded within are themselves B-spline volumes, deformed by displacing the control points on their surface. The control points are arranged into cross-sections which provide local control through three degrees of freedom (DOFs) illustrated in Figure 2: twist, a bulk rotation of the cross-section about the local origin; taper, a bulk scaling of the chord with fixed t/c; and section, individual movement of cross-sectional control points in the local vertical direction. Global control is provided by an axial curve [14], a B-spline curve which determines the orientation of the FFD cross-sections. The axial curve itself is controlled with its own set of control points, each having up to three DOFs: sweep (translation in the flow axis) span (translation in the span axis) and dihedral (translation in the vertical direction). Figure 3 demonstrates global deformation for a simple rectangular wing. Red spheres represent the embedded surface patch control points, black cubes are the FFD control points, and blue spheres are the axial control points. Starting from the undeformed geometry in Figure 3a, sweep is added by pulling the outboard control points back as per Figure 3b; as the axial curve is deformed, each cross-section is translated or rotated to maintain its position and orientation relative to the curve, in turn deforming the FFD volume, embedded surface control points, and then the surface itself.

Finally, a note on nomenclature employed in this work. The FFD system consists of three modes of deformation corresponding to the three types of structure that the optimizer may manipulate: axial control points, cross-sectional control points, and cross-sections (which, while composed of cross-sectional control points, can be deformed independently of the individual points). Each mode of deformation may have one or more DOFs: cross-sectional control points may only have a single DOF (section shape), cross-sections may have up to two (twist and taper), and axial control points may have up to three (sweep, span, and dihedral). Design variables (DVs) refer to the individual values the optimizer may manipulate during optimization; a single DOF may encompass multiple DVs.



(a) Twist control (b) Taper control (c) Section control

Fig. 2 Cross-sectional design variables



(a) Undeformed (b) Deformed

Fig. 3 Axial deformation

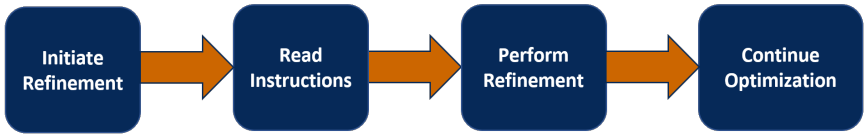
III. Dynamic Geometry Control

The overall structures of the progressive and adaptive algorithms are laid out in Figure 4. In both cases refinement is initiated by a refinement criterion which is designed to detect when the current search space (the design space defined by the current geometry control scheme) has been exhausted. From there the progressive algorithm is straightforward: the user-defined instructions are read in from a file, the desired points are added to the geometry control scheme, and optimization proceeds. This process is repeated a pre-determined number of times, and once the final requested refinement has been completed, the optimizer will iterate in the current search space until it exits or the user requests termination.

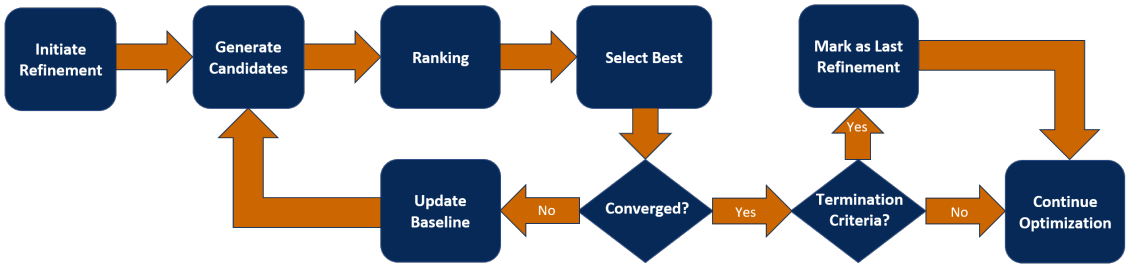
The adaptive algorithm is considerably more involved. Once refinement is initiated, the algorithm generates a number of candidate search spaces, each one a particular refinement of the current geometry control scheme. Each of these candidates is then ranked based on an estimate of its potential, which can be obtained in several ways, and the best candidate is accepted. Two termination criteria, discussed below, are used to control the behaviour of the adaptive algorithm. The local termination criterion is checked after each candidate is accepted and determines whether to continue refining or proceed with optimization. The global criterion is checked after the local criterion is satisfied and controls whether this will be the final search space to explore. In the adaptive algorithm there is not generally a manual limit placed on the number of times refinement may occur, and optimization will proceed in this cycle of optimization-exhaustion-refinement until the global termination criterion is met, at which point optimization proceeds in the final search space until the user is satisfied that all optimization convergence criteria have been satisfied. Salient details of each of these stages are provided below.

A. Initiating and Terminating Refinement

One of the largest considerations in designing a DGC algorithm is when to refine the design space. Several different approaches are present in the literature. Han and Zingg [47] converged the problem fully within each search space, whereas Anderson and Aftosmis [50] initiate refinement when objective improvement has become asymptotic. Masters et al. [46] use a similar method, but employ several rolling averages to smooth out the refinement switch and include both the objective and constraints in the formulation. Anderson and Aftosmis showed that fully converging, as in Han and Zingg, tended to be less efficient than their method, and internal testing by the present authors suggests Anderson and Aftosmis' method produces consistent results and is also easy to tune as desired; all tests presented results utilize a variation of this slope criterion.



(a) Progressive algorithm



(b) Adaptive algorithm

Fig. 4 Dynamic geometry control algorithms

The refinement criterion implemented here signals that the current search space is exhausted, and therefore refinement is due, when:

$$\begin{aligned} \frac{\Delta\mathcal{M}}{\Delta\mathcal{M}_{\text{locmax}}} &\leq r_{\text{loc}} \text{ for } n_{\text{loc}} \text{ consecutive iterations, or} \\ \frac{\Delta\mathcal{M}}{\Delta\mathcal{M}_{\text{glbmax}}} &\leq r_{\text{glb}} \text{ for } n_{\text{glb}} \text{ consecutive iterations} \end{aligned} \tag{1}$$

where $\Delta\mathcal{M}$ is the merit function reduction between the last two design iterations, $\Delta\mathcal{M}_{\text{max}}$ is the largest reduction seen since previous refinement (locmax) or the beginning of optimization (glbmax), and the r and n parameters are user-defined. Merit function is preferred over the objective as merit is generally monotonically reducing and so makes evaluation of these criteria more straightforward. The dual criteria augment Anderson's criterion by separating it into local and global components. The local criterion is meant to initiate refinement when the current search space has been exhausted, while the global criterion protects against the optimizer becoming trapped in a poor search space that never produces an appreciable reduction in the objective, and so may fail to make sufficient progress but never trigger the local criterion. In practise, the former criterion dominates in early search spaces while the latter criterion is often more active later in the design process when merit gains are more likely to be marginal.

Once refinement has begun, some metric must be used to determine when to stop and proceed with optimization. For progressive geometry control this is quite simple: refinement proceeds until all requested control has been added to the search space. For adaptive geometry control, refinement is terminated when:

$$\frac{(\mathcal{A}_i - \mathcal{A}_{\text{BL}})}{\mathcal{A}_{\text{max}}} \leq a_{\text{loc}} \text{ for } m_{\text{loc}} \text{ consecutive candidates,} \tag{2}$$

where $\mathcal{A}_i$ is the potential of the i^{th} candidate to be added to the design space, $\mathcal{A}_{\text{BL}}$ is the potential of the design space without $\mathcal{A}_i$, $\mathcal{A}_{\text{max}}$ is the best potential found so far during the current refinement, and $0 < a_{\text{loc}} < 1$ and $m_{\text{loc}} > 0$ are user-defined parameters. The values of $\mathcal{A}$ are estimates of the objective reduction that a given design space would permit beyond the current point; this is the metric by which the adaptive candidates are ranked and their calculation is discussed in Section III.B. This criterion then ends refinement when the expected benefits of further refinement drop below some user-defined threshold. When the termination criterion in equation 2 is met, a second "global" refinement termination criterion is checked as well. This determines whether the optimization problem has converged with respect to refinement. The global criterion is tripped if

$$\frac{\mathcal{M}_{\text{init}} - \mathcal{M}_{\text{final}}}{\mathcal{M}_{\text{init}}} \leq a_{\text{glb}} \text{ for } m_{\text{glb}} \text{ consecutive search spaces} \tag{3}$$

where $\mathcal{M}_{\text{init}}$ is the merit function at the first iteration in the search space, $\mathcal{M}_{\text{final}}$ is the merit function at refinement, and $0 < a_{\text{glb}} < 1$ and $m_{\text{glb}} > 0$ are user-defined parameters. This criterion states that if the total reduction in merit across the

Table 1 Typical Values for Refinement Parameters

Parameter	Typical Range
r_{loc}	0.05 - 0.10
r_{glb}	0.005 - 0.01
n_{loc}	5-10
n_{glb}	25-50
a_{loc}	0.001 - 0.10
a_{glb}	0.005
m_{loc}	2
m_{glb}	3

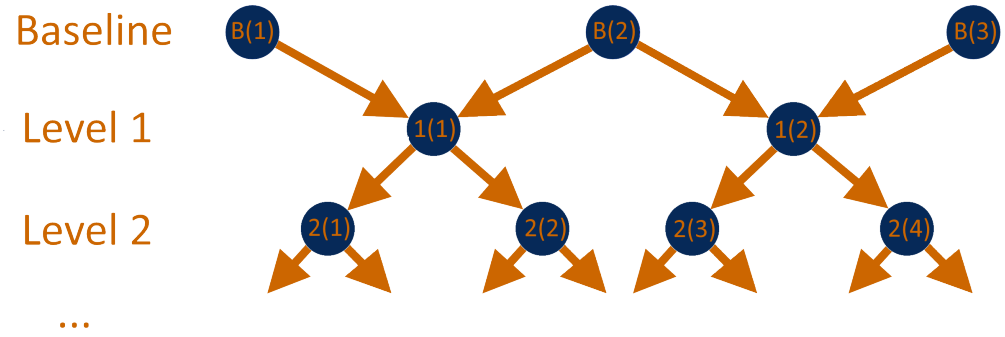


Fig. 5 Candidate generation binary search tree

last few design spaces is below some user-defined percentage, then the problem has converged. If satisfied, optimization is not terminated, but rather a note is made to prevent any further refinements and the optimizer is permitted to run in the current design space until convergence criteria are satisfied, SNOPT exits of its own accord, or the user terminates the job.

Typical values for the parameters in equations 1, 2, and 3 are given in Table 1. Tests indicated that DGC performance is largely insensitive to values of r, and optimal ranges of this value are not highly problem dependent. On the other hand, while adaptive performance was found to be fairly stable with respect to the value of a_{loc} for detailed-design problems, sensitivity is much higher for problems more typical of preliminary and exploratory design. These design spaces have previously been noted as having a high risk of multimodality [26] and it is possible that this sensitivity is in fact an expression of multimodality within the problem. This could have major implications for the use of adaptive DGC in exploratory optimization, but further investigation must await a future work.

B. Candidate Generation and Ranking

For adaptive geometry control we must generate a list of possible refinements and then formulate some way to rank their usefulness. This is achieved through the use of a binary search tree, as illustrated in Figure 5. The baseline level is composed of a list of all existing control points and each lower level is then formed by inserting a new control point halfway between the control points at the previous level (while retaining all existing points). In this way a finite number of candidates can be produced at any level, while still approaching a continuous geometry control scheme as the depth goes to infinity. In our refinement scheme, no candidate can be considered until at least one predecessor control point has been added, for instance in Figure 5, candidates 2(1) and 2(2) could not be considered until candidate 1(1) was added.

Once the candidates are generated each one is ranked, the best candidate is added to the baseline, and the process is repeated with the updated baseline until the exit criterion is satisfied. By default, the algorithm may only examine a

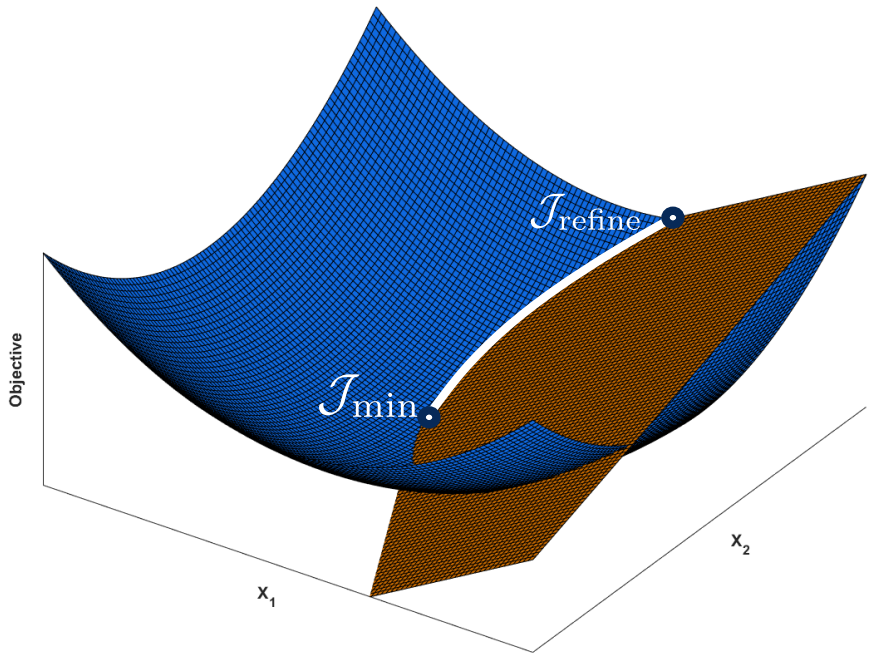


Fig. 6 Quadratic fit of candidate design space. Quadratic objective in blue, linearized constraints in orange

single control point at a time, in which case in Figure 5 the first stage of refinement would see only two candidates: 1(1) and 1(2). Suppose that 1(2) is then added; assuming that the exit criterion is not satisfied, a second stage of refinement is started in which the algorithm would now see three candidates: 1(1), 2(3), and 2(4), with the latter two now being available by virtue of the addition of their predecessor 1(2).

For a given stage of refinement, evaluation of the binary search tree will create a list of dozens or even hundreds of candidates which must be ranked. This ranking is performed on the basis of "potential", which is expressed in the value of the candidate "indicator". The indicator is an estimate of the maximum objective reduction that a given candidate refinement would permit relative to the current objective. We obtain this estimate via a quadratic fit of each candidate design space subject to a linearized set of constraints, this forms a constrained quadratic minimization problem, the solution of which is an estimate of the minimum feasible objective in the search space. This is illustrated in Figure 6 for a simple case with just two design variables; the indicator for this candidate is the difference between the current objective ($\mathcal{J}_{\text{refine}}$) and the objective at the minimum of the constrained quadratic fit ($\mathcal{J}_{\text{min}}$). The lower $\mathcal{J}_{\text{min}}$ is for a candidate, the greater the reduction in objective this candidate theoretically permits. The optimization problem we solve for each candidate can be formally stated as

$$\begin{aligned} \text{minimize} \quad & \Delta\mathcal{J} = \left(\frac{\partial\mathcal{J}}{\partial\widehat{\mathbf{X}}}\cdot\Delta\widehat{\mathbf{X}}\right) + \frac{1}{2}\left(\Delta\widehat{\mathbf{X}}\cdot\widehat{H}\Delta\widehat{\mathbf{X}}\right) \\ \text{Subject to} \quad & C^a_{0,i} + \left(\frac{\partial C^a_i}{\partial\widehat{\mathbf{X}}}\cdot\Delta\widehat{X}\right) = 0, \end{aligned} \tag{4}$$

where $\mathcal{J}$ is the objective, $\widehat{\mathbf{X}}$ are the design variables in the candidate search space, $\widehat{H}$ is the Hessian of that search space, and C^a are the active constraints. Note that C^a will often contain a mix of equality and inequality constraints, but as they are all active they may be treated uniformly as equality constraints. If the candidate permits a further decrease in objective, we expect $\Delta\mathcal{J}$ to be a negative value and so we define our indicator as $\mathcal{A} = -\Delta\mathcal{J}$. As a constrained quadratic optimization problem, finding the minimum for a given set of values is not a taxing problem. However, forming each system requires us to obtain three key pieces of data: 1) $\partial\mathcal{F}/\partial\widehat{\mathbf{X}}$, gradients of the objective and each constraint in the

candidate search space, 2) C^a, the set of active constraints, and 3) $\widehat{H}$, the objective Hessian in the candidate search space.

1. Gradient Calculation

Gradients can be obtained cheaply and at high accuracy by noting that in Jetstream the gradient of each function $\mathcal{F}$ is calculated as

$$\frac{\partial \mathcal{F}}{\partial \mathbf{X}} = \frac{\partial \mathcal{F}}{\partial \mathbf{b}} \frac{\partial \mathbf{b}}{\partial \mathbf{X}}, \tag{5}$$

where $\mathbf{b}$ are the embedded surface control points, $\mathbf{X}$ are the FFD design variables, and $\mathcal{F}$ is any function. The expensive components of the gradient calculation, including the adjoint solutions, are contained within $\partial \mathcal{F}/\partial \mathbf{b}$, while $\partial \mathbf{b}/\partial \mathbf{X}$ is purely geometric and may be calculated analytically or estimated cheaply and to arbitrary accuracy with the complex step method [65]. Refining the FFD invalidates the latter term as $\partial \mathbf{b}/\partial \mathbf{X} \neq \partial \mathbf{b}/\partial \widehat{\mathbf{X}}$; however, $\partial \mathcal{F}/\partial \mathbf{b}$ is a function of the surface parameterization, not the FFD, and so will remain constant if the geometry does not deform during refinement. As we guarantee that the surface does not move during refinement, we can save the embedded gradients from the current search space and for each candidate calculate the overall gradients as

$$\frac{\partial \mathcal{F}}{\partial \widehat{\mathbf{X}}} = \frac{\partial \mathcal{F}}{\partial \mathbf{b}} \frac{\partial \mathbf{b}}{\partial \widehat{\mathbf{X}}}, \tag{6}$$

for only the negligible cost of evaluating $\frac{\partial \mathbf{b}}{\partial \widehat{\mathbf{X}}}$. Even for progressive geometry control, for which candidate's gradients are not needed, preserving the surface is essential to ensuring good performance in the refined search space. To achieve this objective, a combination of knot insertion [66] and re-embedding algorithms is used to ensure that the surface is fixed throughout the refinement process.

2. Defining the Set of Active Constraints

The active set can also be approximated in a cost-effective and uniform manner. In previous quadratic fit methods [51] it was assumed that the active set at the minimum of the quadratic fit was equal to the active set at the point of refinement, i.e. $C^a_{\text{refine}} = C^a_{\text{min}}$. The assumption of a fixed active set has also been made by other authors who have included constraints in their indicators (though without fitting the design space) [49]. This is convenient as it makes the active set simple to obtain and in the case of quadratic indicators allows equation 4 to be solved in a single iteration. However, unless the minimum in the refined search space is very close to the point of refinement it is unlikely that the active sets at both points are equal. Since it can be argued that the purpose of adaptive geometry control is to find a new search space whose minimum is as far as possible from the current point, it is likely that this assumption is invalid.

This issue can be addressed by noting that the linearized constraints enforced in equation 4 allow us to estimate what the value of any constraint, not merely those already active, would be at a new point in the design space, permitting the use of an active set method. The active set method, as shown in Figure 7, consists of solving equation 4 subject to a given set of constraints, updating the active set to add violated constraints or remove unnecessary ones, solving the problem again subject to the new active set, and repeating the process until there are no more constraints to add or remove. This requires that equation 4 be solved multiple times, but this is a negligible cost and permits a greater degree of accuracy in the ranking of each candidate.

3. Hessian Formulation

The final outstanding issue in the evaluation of the quadratic fit is the formulation of $\widehat{H}$, the objective Hessian in the candidate search space. Obtaining accurate Hessian data is not straightforward and has significant influence over the behaviour of the adaptive algorithm. Four Hessian approximations are examined, and these form the basis of the four adaptive algorithms tested in this work.

The simplest approach, which was examined by Anderson and Aftosmis [50], is to assume that the Hessian is the identity matrix. This has the advantage of solely using cheap and accurate first-order sensitivity data, but without making the unrealistic assumption that the objective of an aerodynamic shape optimization problem is linear. We denote this indicator as $\mathcal{A}_{\text{QI}}$ and it is obtained by simplifying equation 4 to

$$-\mathcal{A}_{\text{QI}} := \begin{cases} \text{minimum} & \left(\frac{\partial \mathcal{J}}{\partial \widehat{\mathbf{X}}} \cdot \Delta\widehat{\mathbf{X}}\right) + \frac{1}{2}\left(\Delta\widehat{\mathbf{X}} \cdot \Delta\widehat{\mathbf{X}}\right) \\ \text{subject to} & C^a_{0,i} + \left(\frac{\partial C^a_i}{\partial \widehat{\mathbf{X}}} \cdot \Delta\widehat{\mathbf{X}}\right) = 0 \end{cases} \tag{7}$$

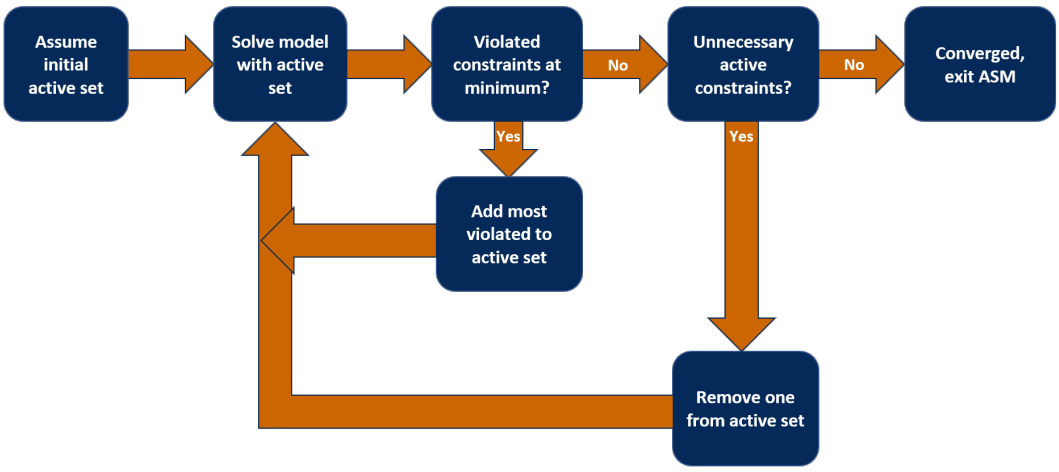


Fig. 7 Active set method

This must be done with caution, as assuming that $\widehat{H} = I$ is not removing the Hessian from consideration, but merely assuming a specific set of values for it. Assuming $\widehat{H} = I$ is no more valid than any other arbitrary SPD matrix and a curvature of 1 will produce very different behaviour in the indicator depending on the scaling of the problem. Therefore, it is advisable to scale the identity matrix to more closely match the scale of the problem at hand. If refinement is occuring after the i^{th} design iteration, we can scale the identity Hessian as

$$\begin{aligned} H &= \nu I \\ \nu &= \sqrt{\frac{\sum_j \left(\frac{\partial \mathcal{J}^{(i)}}{\partial \mathbf{X}_j} - \frac{\partial \mathcal{J}^{(i-1)}}{\partial \mathbf{X}_j} \right)^2}{\sum_j \left(\mathbf{X}_j^{(i)} - \mathbf{X}_j^{(i-1)} \right)^2}} \end{aligned} \tag{8}$$

where I is the identity matrix and $\mathbf{X}_j^{(i)}$ is the j^{th} entry of the design variable vector at the i^{th} design iteration. This scaling factor is commonly used to initialize iterative Hessian approximations such as the BFGS method [67] and should be sufficient to ensure that the Hessian is approximately of the correct order of magnitude. Substituting this into equation 4 and simplifying, we find the equation for the scaled identity indicator, denoted as $\mathcal{A}_{\text{QIS}}$, is:

$$-\mathcal{A}_{\text{QIS}} := \begin{cases} \text{minimum} & \left(\frac{\partial \mathcal{J}}{\partial \widehat{\mathbf{X}}} \cdot \Delta \widehat{\mathbf{X}} \right) + \frac{\nu}{2} \left(\Delta \widehat{\mathbf{X}} \cdot \Delta \widehat{\mathbf{X}} \right) \\ \text{subject to} & C_{0,i}^a + \left(\frac{\partial C_i^a}{\partial \widehat{\mathbf{X}}} \cdot \Delta \widehat{\mathbf{X}} \right) = 0 \end{cases} \tag{9}$$

The identity matrix approach is simple and inexpensive, but of low accuracy. Many gradient-based optimization algorithms build a Hessian approximation as part of their operation which could be leveraged for the quadratic fit. However, not all algorithms permit the user to extract the Hessian and so a more general approach to building higher-accuracy Hessian approximations is desirable. Later work by Anderson [51] suggests the use of a BFGS approximation of the Hessian, achieved by first obtaining a BFGS approximation of the Hessian in the current search space (H) and then projecting this approximation into each candidate search space to locate $\widehat{H}$. The downside of this method is that the prolongation operator to project the Hessian into the new search space is dependent on the objective function used, and for easy portability an objective-agnostic Hessian approximation would be preferred.

For this reason we have developed two alternative Hessian approximations which do not require the objective-specific prolongation operators employed by Anderson, one based on the BFGS approximation and the other the Symmetric Rank-1 (SR1) [67] approximation. The BFGS approach is ideal for optimization, as when implemented as part of a quasi-Newton optimization algorithm it guarantees an SPD Hessian matrix. The SR1 method has no such guarantee, but theoretically offers superior accuracy compared to the BFGS approximation. The approximations themselves are

commonly written as

$$
\begin{aligned}
H_{\mathrm{B}}^{(i+1)} &= H_{\mathrm{B}}^{(i)} + \frac{\mathbf{y}\mathbf{y}^{\intercal}}{\mathbf{y}^{\intercal}\mathbf{s}} - \frac{H_{\mathrm{B}}^{(i)}\mathbf{s}\mathbf{s}^{\intercal}\left(H_{\mathrm{B}}^{(i)}\right)^{\intercal}}{\mathbf{s}^{\intercal}H_{\mathrm{B}}^{(i)}\mathbf{s}} \\
H_{\mathrm{R}}^{(i+1)} &= H_{\mathrm{R}}^{(i)} + \frac{\left(\mathbf{y} - H_{\mathrm{R}}^{(i)}\mathbf{s}\right)\left(\mathbf{y} - H_{\mathrm{R}}^{(i)}\mathbf{s}\right)^{\intercal}}{\left(\mathbf{y} - H_{\mathrm{R}}^{(i)}\mathbf{s}\right)^{\intercal}\mathbf{s}} \\
\mathbf{s} &= \mathbf{X}^{(i+1)} - \mathbf{X}^{(i)} \\
\mathbf{y} &= \nabla\mathcal{F}^{(i+1)} - \nabla\mathcal{F}^{(i)}
\end{aligned}
\tag{10}
$$

where H_{B} is the BFGS approximation, H_{R} is the SR1 approximation, $\mathbf{X}^{(i)}$ is the design variable vector at the i^{th} iteration and $\mathcal{F}$ is the function whose Hessian is being approximated. Substituting these Hessian approximations into equation 4 yields the equations for the last two indicators examined in this work:

$$
-\mathcal{A}_{\mathrm{QB}} := \begin{cases} \text{minimum } \Delta\mathcal{J} = \left(\frac{\partial\mathcal{J}}{\partial\widehat{\mathbf{X}}}\cdot\Delta\widehat{\mathbf{X}}\right) + \frac{1}{2}\left(\Delta\widehat{\mathbf{X}}\cdot\widehat{H}_{\mathrm{B}}\Delta\widehat{\mathbf{X}}\right) \\ \text{subject to} \quad C_{0,i}^{a} + \left(\frac{\partial C_i^a}{\partial\widehat{\mathbf{X}}}\cdot\Delta\widehat{\mathbf{X}}\right) = 0 \end{cases} \tag{11}
$$

$$
-\mathcal{A}_{\mathrm{QR}} := \begin{cases} \text{minimum } \Delta\mathcal{J} = \left(\frac{\partial\mathcal{J}}{\partial\widehat{\mathbf{X}}}\cdot\Delta\widehat{\mathbf{X}}\right) + \frac{1}{2}\left(\Delta\widehat{\mathbf{X}}\cdot\widehat{H}_{\mathrm{R}}\Delta\widehat{\mathbf{X}}\right) \\ \text{subject to} \quad C_{0,i}^{a} + \left(\frac{\partial C_i^a}{\partial\widehat{\mathbf{X}}}\cdot\Delta\widehat{\mathbf{X}}\right) = 0 \end{cases} \tag{12}
$$

Our approach differs from that of Anderson in that rather than applying the algorithms in the current search space and then projecting the Hessian into each candidate search, we instead seek to directly obtain an approximate Hessian in each candidate search space by applying the algorithm within it. To accomplish this, during optimization the surface gradient of each function $\partial\mathcal{F}^{(i)}/\partial\mathbf{b}$ is saved. Once refinement has been triggered and a given candidate created by refining the geometry control, at each of these iterations the algorithm solves the geometric shape-matching subproblem

$$
\underset{\widehat{\mathbf{X}}^{(i)}}{\text{minimize}} \quad \left[\mathcal{S}(\mathbf{X}^{(i)}) - \hat{\mathcal{S}}(\widehat{\mathbf{X}}^{(i)})\right]^2, \tag{13}
$$

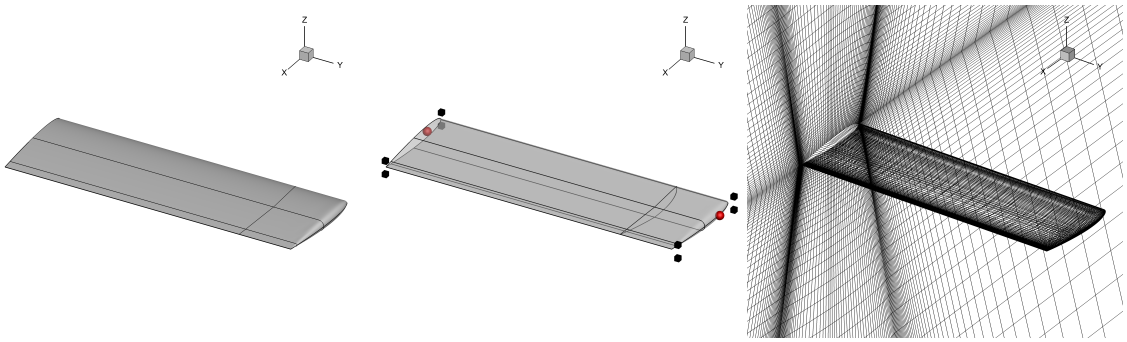
where $\mathcal{S}$ and $\mathbf{X}^{(i)}$ are the surface and design variables in the unrefined search space at the i^{th} iteration, and $\hat{\mathcal{S}}$ and $\widehat{\mathbf{X}}^{(i)}$ are the corresponding values in the candidate search space. This is a purely geometric minimization problem and with proper parallelization is not a significant cost. The result of this solution is a design variable vector $\widehat{\mathbf{X}}^{*}$ in the candidate search space which produces the same surface, and therefore same aerodynamic solution, as was produced at the i^{th} iteration in the unrefined search space. Once this has been located, equation 6 is leveraged to obtain $\partial\mathcal{J}/\partial\widehat{\mathbf{X}}^{(i)}$ and the resulting design variable vector and gradient pair are fed into equation 10, producing an approximation of the Hessian in the candidate search space. This is in effect retroactively applying the BFGS and SR1 Hessian approximations to the candidate design space as if it had always been in use, and for this reason we refer to this approach as the "retroactive Hessian" method; the advantage of this approach is that it is equally applicable to any objective function or flow regime.

IV. Results

With the details of the progressive and adaptive algorithms established, we will now turn to several cases testing and illustrating their performance. The first two cases are inviscid three-dimensional optimization problems focused on testing the adaptive indicators. The last two cases are RANS-based aerodynamic shape optimization problems, one representative of detailed design and the other representative of optimization problems typical of unconventional aircraft configuration research; for these cases we examine the behaviour of static geometry control and the performance of both progressive and adaptive DGC.

A. Adaptive Indicator Studies

The purpose of the following study is to directly test the accuracy of the indicators. Two cases are attempted, each based on the inviscid, transonic optimization of an initially unswept, planar wing with a NACA 0012 cross-section. Both cases use the same baseline geometry control and mesh, depicted in Figure 8, and take drag as the objective function.



(a) Baseline geometry **(b) Baseline geometry control** **(c) Baseline mesh**

Fig. 8 Baseline geometry, geometry control and mesh for full scale indicator verification

The baseline geometry control is intentionally very sparse, and the baseline wing in transonic flow begins with a large shock over the upper surface which is difficult to eliminate, providing plenty of room for improvement through DGC. For each case, the baseline design space is optimized until the refinement criterion is satisfied and adaptive enrichment initiated. At this point, each candidate has its potential estimated using all four indicators before being optimized to completion to obtain its true potential, allowing each indicator to be correlated to actual potential. A quadratic fit, even one with highly accurate Hessian information, is a significant simplication of the real design space and so we are not necessarily expecting to accurately predict the magnitude of each potential, but this is not necessary for the indicator to be effective. If we can achieve reasonable accuracy in ranking the potential of the candidate search spaces, then the indicator should be successful in guiding refinement. Additionally, to test the value of the active set method, each indicator is tested both with and without the active set method; in the latter case, the current active set is used for the solution.

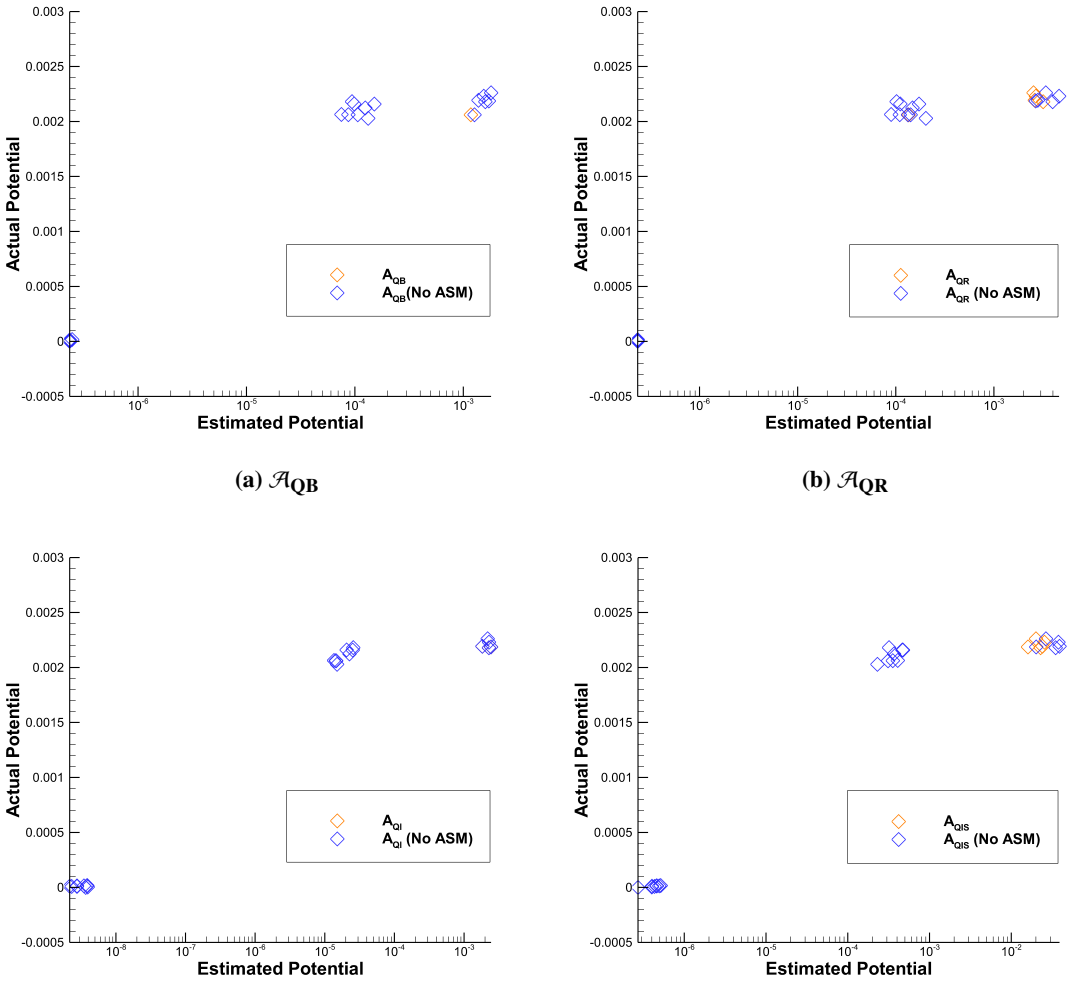
The first of the two cases permits only changes to section shape and is defined as

$$\begin{aligned} \text{minimize} \quad & C_D \\ \text{w.r.t.} \quad & \mathbf{X} \\ \text{subject to} \quad & C_L = 0.1 \\ & V \geq V_{\text{init}} \end{aligned} \tag{14}$$

where V is the total volume and an additional linear constraint requires that section thickness at no point be less than 50% of the initial value. This case was designed so that there would be clear winners and losers among the candidate refinements; each indicator must be able to reliably differentiate between these two families of candidates, otherwise it is unlikely to be useful in practice. Figure 9 shows the resulting correlations.

Examining the vertical distribution of points we can see that we have obtained the desired distribution of actual potential, with one family of near-zero potential candidates clustered about the origin and another family of competitive candidates clustered much further up. All four indicators are effective at differentiating between the two families, without any competitive candidates ranked as non-competitive or vice-versa. However, the indicators consistently introduce a third family by splitting the competitive family in two. This tendency to smear high-potential candidates and exaggerate their differences in performance will be noted in the following case as well and suggests the indicators may be oversensitive to small changes in these design spaces. Nevertheless, this does not prevent them from producing accurate rankings, and within the competitive family, the indicator value correlates well with actual potential. Hence, any of these indicators would permit the adaptive algorithm to select promising candidates.

To study the impact of the active set method, results without the active set method (blue) are superimposed on the results with the active set method (orange), therefore in cases where the active set method has no impact only the blue points will be visible and the overall activity of the active set method can be deduced at a glance from the number of visible orange points. For the $\mathcal{A}_{\text{QB}}$ and $\mathcal{A}_{\text{QI}}$ indicators, the active set method has a negligible effect, while for the $\mathcal{A}_{\text{QR}}$ and $\mathcal{A}_{\text{QIS}}$ indicators it has a much greater visibility, in particular for the rightmost candidates (those with greatest potential). This is intuitive as a large potential (all else being equal) suggests a larger step to the minimum, and a greater chance that additional constraints become active. While in this case the active set method does not significantly change candidate rankings its increased relevance to higher-potential candidates is interesting, as these are the candidates the



(a) $\mathcal{A}_{\mathbf{QB}}$ **(b)** $\mathcal{A}_{\mathbf{QR}}$

(c) $\mathcal{A}_{\mathbf{QI}}$ **(d)** $\mathcal{A}_{\mathbf{QIS}}$

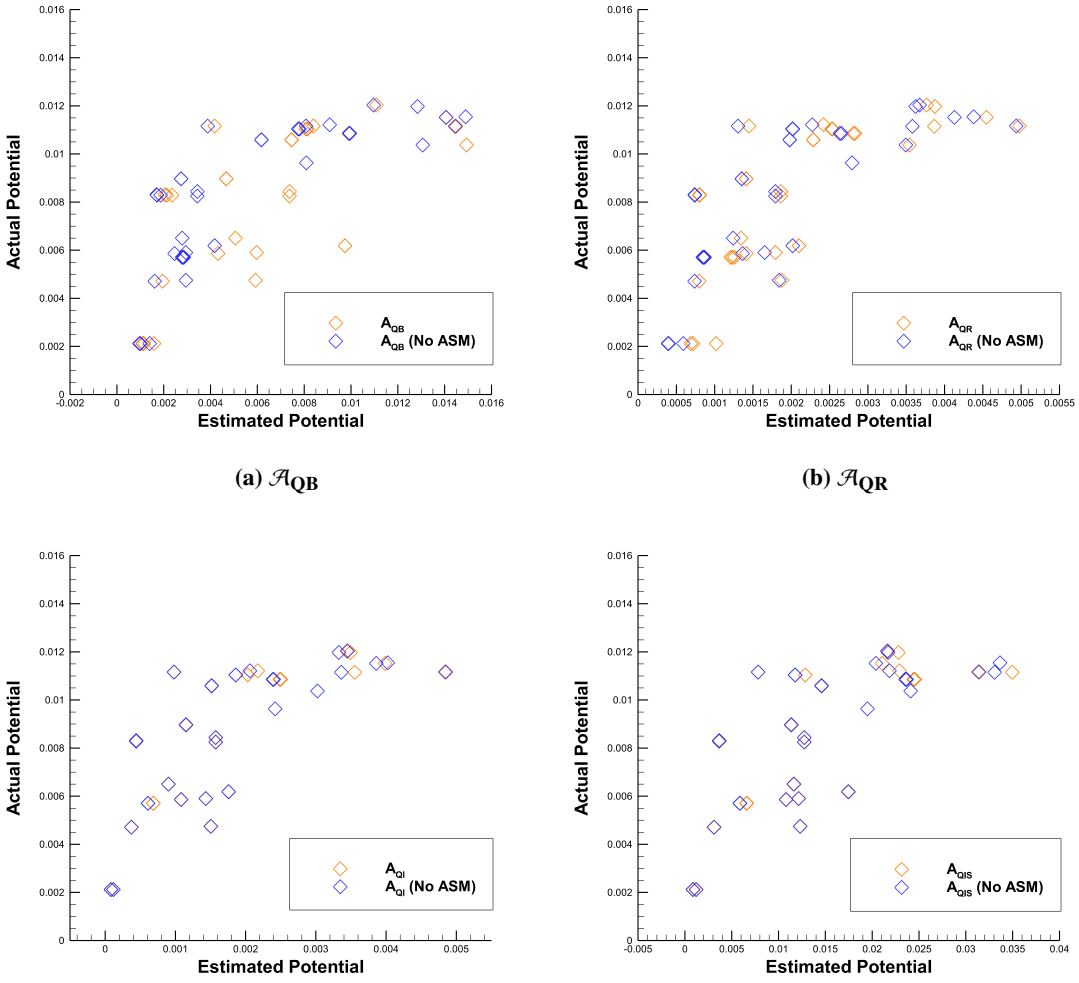
Fig. 9 Predicted vs. actual candidate search space potential for section-only inviscid case

algorithm is most likely to select and those that we most want to accurately model. Finally, it is telling how similar the correlations are for all of the fitted indicators; in this particular design space a less advanced Hessian formulation does not produce a degradation in accuracy.

The second inviscid optimization problem is an exploratory lift-constrained drag minimization permitting changes to twist, taper, section shape, and sweep angle meant to test the behaviour of the indicators in a more complicated optimization problem. This case uses the same baseline geometry, geometry control, and mesh as the previous case and is formally defined as

$$\begin{aligned} \text{minimize} \quad & C_D \\ \text{w.r.t.} \quad & \mathbf{X} \\ \text{subject to} \quad & C_L = 0.1 \\ & V \geq V_{\text{init}} \\ & S = S_{\text{init}} \end{aligned} \tag{15}$$

where S is the projected area and V is the total volume. The minimum thickness constraint from before is retained, as is the test procedure, and the resulting correlations are provided in Figure 10. The correlations are somewhat noisier



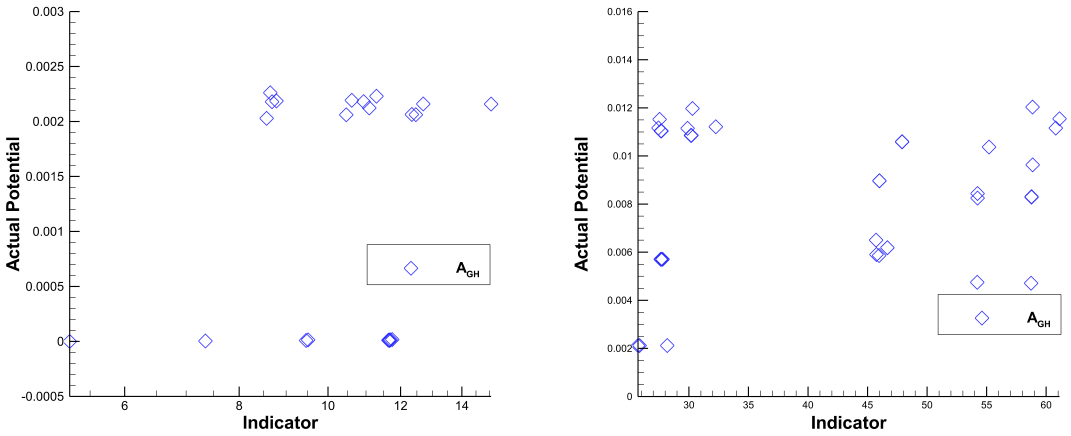
(a) $\mathcal{A}_{\mathbf{QB}}$ **(b)** $\mathcal{A}_{\mathbf{QR}}$

(c) $\mathcal{A}_{\mathbf{QI}}$ **(d)** $\mathcal{A}_{\mathbf{QIS}}$

Fig. 10 Predicted vs. actual candidate search space potential for exploratory inviscid case

than previously, not surprisingly for a more complicated problem, but we still observe a good correlation between indicators and actual potential. As before the indicator appears to become more sensitive for candidates with larger actual potentials, shown by a scatterplot that is predominantly vertical to the left and increasingly horizontal towards the right. The combination of additional constraints and generally larger potentials for each candidate has made the active set method much more visible in this case; in particular for the $\mathcal{A}_{\text{QB}}$ and $\mathcal{A}_{\text{QR}}$ indicators deactivating the active set method significantly changes both the indicator values and their relative rankings. We also again see no sizeable performance gap between the nominally more accurate $\mathcal{A}_{\text{QB}}$ or $\mathcal{A}_{\text{QR}}$ indicators and the more basic $\mathcal{A}_{\text{QIS}}$, lending further credence to the theory that the value of accurate Hessian data may be less than expected.

These plots show that good performance can be achieved by assuming a static active set. Nevertheless, we believe the active set method to be advantageous as it is most active in the candidates of greatest interest and automatically vanishes in cases where it is of less importance. Therefore, all further cases are run with the active set method active. This study has also clouded the potential advantages of Hessian information by showing largely similar performance regardless of the accuracy of the Hessian data used in constructing the fit. Despite the similarity in the performance of the various Hessian approximations, the use of a quadratic model of the design space is nevertheless important. In order to demonstrate this, we repeat the above tests using a "non-fitting" indicator, that is an indicator which does not attempt to model the shape of the design space. The indicator chosen for this test was developed by He et al.[49], which we



(a) Section-only case **(b) Exploratory case**

Fig. 11 $\mathcal{A}_{\mathrm{GH}}$ accuracy

denote as $\mathcal{A}_{\mathrm{GH}}$ and is defined as

$$\mathcal{A}_{\mathrm{GH}} := \sum_{i=1}^{n_y} \left(\frac{\partial \mathcal{J}}{\partial \widehat{X}_i} - \sum_{j=1}^{n_g} \frac{\partial C_j^a}{\partial \widehat{X}_i} \right)^2 . \tag{16}$$

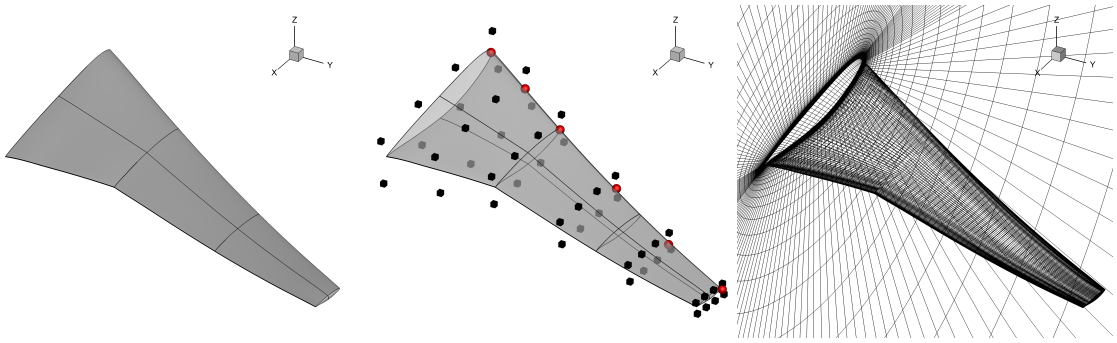
This ranks each candidate based on the value of the objective gradient, penalized by the gradients of any active constraints. This formulation accounts for the fact that while a large $|\partial \mathcal{J}/\partial X_i|$ value suggests high potential, an active constraint with a large $|\partial C/\partial X_i|$ value of the same sign may block any attempt to move a long distance in the optimal direction, and is a considerable improvement over more basic non-fitting indicators like the earlier work of Han and Zingg [47], which ranked solely based on objective gradient and relied on heuristics to account for constraints. Nevertheless, when this indicator is applied to the two previously discussed cases, as in Figure 11, we see that it struggles to reliably predict potential. Non-fitting indicators have been used to produce reasonable results before [47, 49], and these methods may still produce performance improvement versus static design spaces, however, they are likely to be less efficient or reliable than fitting indicators. The present results indicate that while using accurate Hessian data in the quadratic fit does not impart significant advantage to the indicator, the quadratic fit itself has value in modelling the behaviour of a candidate design space.

B. Transonic, Viscous Wing Optimization

We now study the first of two RANS aerodynamic shape optimization problems in this work. This is the twist and section optimization of the ADODG CRM wing-only geometry in transonic flow subject to the RANS equations, a standard benchmark problem defined as

$$\begin{aligned} \text{minimize} \quad & C_D \\ \text{w.r.t.} \quad & \mathbf{X} \\ \text{subject to} \quad & C_L = 0.5 \\ & V \geq 0.2617\,\mathrm{MAC}^3 \\ & C_M \geq -0.17 \end{aligned} \tag{17}$$

where C_D is the drag coefficient, C_L is the lift coefficient, V the wing volume, and C_M the pitching moment coefficient. A minimum thickness constraint is enforced requiring that the sectional thickness at no point be less than 85% of the initial value. The baseline geometry, geometry control system, and mesh are illustrated in Figure 12 with mesh parameters provided in Table 2. The S0 baseline geometry control in Figure 12b is somewhat coarser than what would



(a) Baseline geometry **(b) Baseline (S0) geometry control** **(c) Baseline mesh**

Fig. 12 Baseline geometry, geometry control, and mesh for CRM

Table 2 CRM mesh parameters

Nodes	1,068,856
Blocks	40
Off-Wall Spacing	7.3×10^{-7} MAC
y^+	0.13

normally be used for this problem, particularly in the chordwise direction, but is nevertheless a fairly reasonable control scheme. A static study was first undertaken to explore how this problem scales with dimensionality by creating a sequence of five progressively finer static schemes, referred to as S0 through S4. Each scheme was created by uniformly refining the previous scheme, inserting a new control point halfway between each existing pair. The resulting series ranges from S0, with a fairly standard 54 design variables, to S4 with 8019 design variables, nearly two orders of magnitude finer than geometry control schemes used in the past. Each of these static schemes was optimized to convergence or failure, at which point the progressive and adaptive DGC algorithms were applied using the parameters in Table 3; the progressive algorithm begins in S0 and moves through each of the static schemes in turn while the adaptive algorithms, also starting in S0, are free to refine as desired.

Merit function convergence histories for all cases are plotted in Figure 13, with optimality and feasibility for the static cases in Figure 14 and the DGC cases in Figure 15. Table 4 summarizes the results for each case, including the number of design iterations until convergence or termination, the maximum number of design variables, and the minimum drag in counts. For unconverged cases the reported number of design iterations is simply how long it was permitted to run before being terminated; for converged cases this value is the number of iterations until merit improvement had planed off and feasibility was satisfied. This is a subjective measure meant to communicate at a glance how quickly a given case converged, but the complete convergence histories can be seen in Figures 13, 14, and 15. Note that the drag values reported in Table 4 are not mesh converged; as the purpose here is to compare performance across different geometry control schemes with identical meshes, the optimization mesh is sufficient.

Regarding the static cases, the observed trends are almost exactly as expected. Refining from S0 through S2, convergence becomes slower and deeper; as refinement continues past S2 into S3 and S4, performance breaks down and achieving convergence becomes increasingly expensive. One unusual finding in this case is that the finest S4 level actually converges somewhat faster than the coarser S3 level. There is no clear reason for this to be the case, but is likely explained by the optimizer managing to find a sequence of good iterates early in the problem, and this does not change the overall noted trends. For clarity all of the convergence plots are truncated after 400 design iterations but here, and wherever else possible, each case was permitted to run until failure or convergence. S4 encountered numerical difficulties and exited after 420 design iterations having found a drag of 201.3 counts, while the S3 was manually terminated after 728 design iterations with a drag of 201.1 counts.

These results illustrate that good convergence is possible for this problem using an ideal static geometry control scheme, e.g. S2, and we now consider the question of whether DGC can match or exceed this performance without a priori knowledge of the design space. The progressive geometry control, plotted as a red dashed line, starts in S0

Table 3 DGC parameters for CRM case

Parameter	Value
r_{loc}	0.05
r_{glb}	0.005
n_{loc}	5
n_{glb}	25
a_{loc}	0.005
a_{glb}	0.005
m_{loc}	2
m_{glb}	3

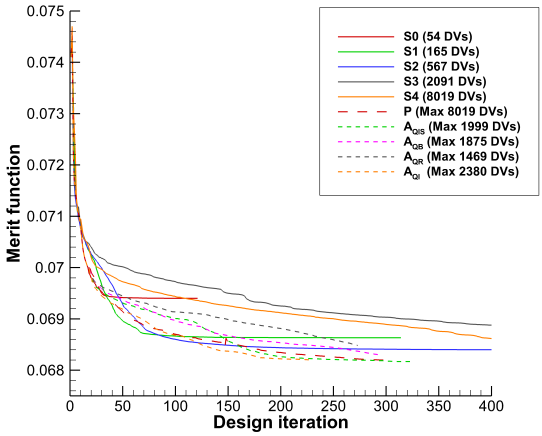
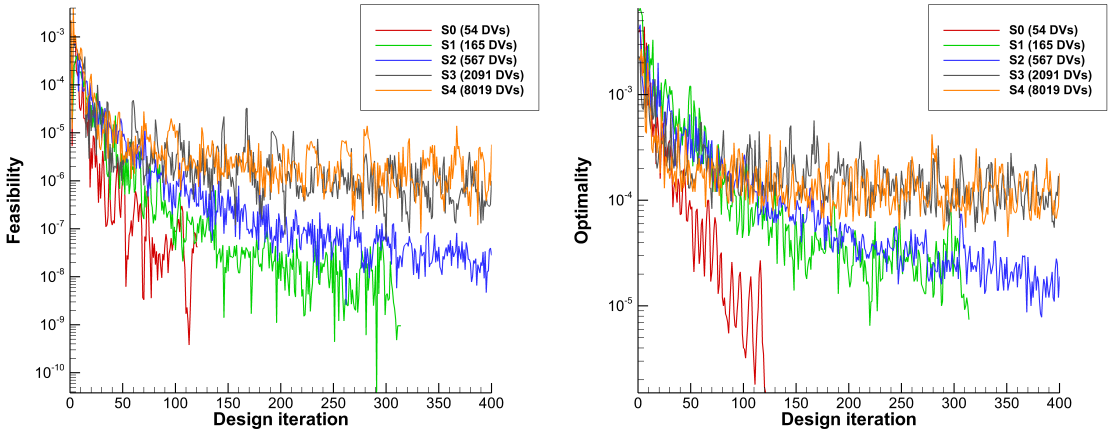


Fig. 13 Merit histories for CRM case

and sequentially moves through each static scheme. This case is able to achieve good convergence, and Figure 13 and Table 4 illustrate that it performs favourably compared to the static schemes. While the progressive case converges in a similar number of iterations as S2, it is able to achieve deeper convergence than any of the static cases within 400 design iterations, producing a final drag roughly a half count lower than S2, the best static case, and over 3.5 counts lower than S0. This is achieved in substantially fewer iterations than the comparably fine S3 and S4 static spaces while simultaneously reducing demands on the user. Achieving the best results via static control would require the user to know that S2, or something like it, is ideal for this problem. However a novice user would be able to achieve even better results with progressive control simply by generating a sequence of geometry control schemes, without any insight required as to which are best or worst performing.

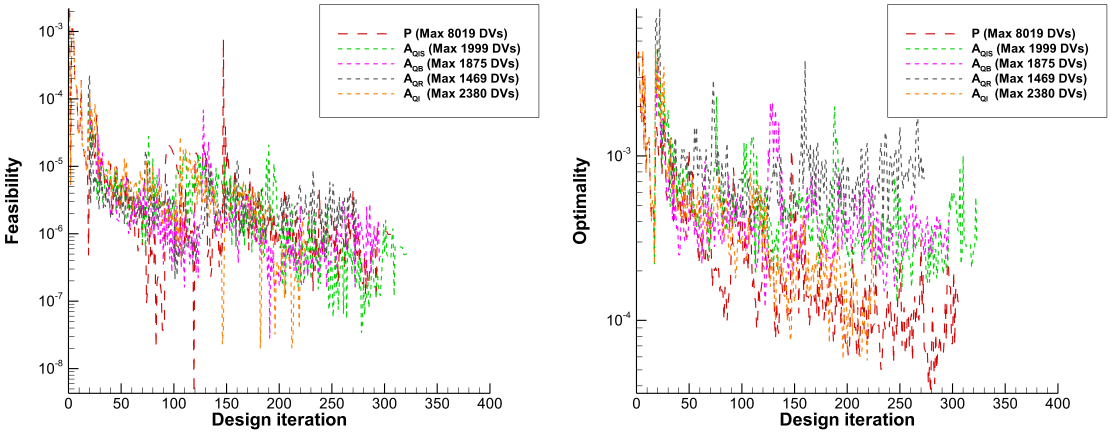
Like progressive geometry control, adaptive control reduces requirements on the user by automating the design of the geometry control, with the potential for further acceleration by focusing control in regions of greatest interest. The $\mathcal{A}_{\text{QIS}}$ and $\mathcal{A}_{\text{QI}}$ adaptive DGC approaches perform particularly well, taking respectively 56 and 106 fewer design iterations than the progressive case to obtain approximately the same drag, and doing so with minimal knowledge requirements on the part of the user. The $\mathcal{A}_{\text{QR}}$ and $\mathcal{A}_{\text{QB}}$ methods perform poorly; while both have found or on track to find lower drag than the fastest static case, S2, their convergence is significantly slower than the other DGC methods.

Some discussion is warranted as to why the indicators based on iterative Hessian approximations perform so poorly compared to the identity-based approaches. The poor performance of $\mathcal{A}_{\text{QR}}$ is not altogether surprising. The use of the SR1 Hessian is a trade-off between its its generally greater accuracy (relative to the BFGS approximation) and its inability to guarantee an SPD Hessian; if increased Hessian accuracy is of minimal advantage (as results thus far have suggested), then we would expect uneven performance from $\mathcal{A}_{\text{QR}}$. The behaviour of $\mathcal{A}_{\text{QB}}$ is interesting, as for the first



(a) Feasibility **(b) Optimality**

Fig. 14 Static convergence for CRM wing optimization



(a) Feasibility **(b) Optimality**

Fig. 15 DGC convergence for CRM wing optimization

150 design iterations the $\mathcal{A}_{\text{QB}}$ and $\mathcal{A}_{\text{QIS}}$ methods shadowed each other before diverging. It is likely relevant that up until this point the number of design variables in both cases were roughly similar, with $\mathcal{A}_{\text{QIS}}$ having 550 DVs and the $\mathcal{A}_{\text{QB}}$ having 605. However, past this point $\mathcal{A}_{\text{QIS}}$ began refining much more aggressively than the BFGS-based indicator, with the former growing the design space 50% faster than the latter over the proceeding 75 iterations. This suggests that the poor performance may be more a matter of refinement timing than the accuracy of the indicator. If so, then minor tweaks to the refinement termination criterion could improve convergence.

The final question relates to the surprisingly good performance of $\mathcal{A}_{\text{QI}}$, which is both the simplest and most effective indicator in this case; the explanation may again be the number of design variables. While it ultimately produced more design variables than any other case, for much of the first 150 design iterations $\mathcal{A}_{\text{QI}}$ produced much coarser search spaces than any other indicator, only deeply refining much later during optimization. Why this would be advantageous here comes down to clustering. This geometry begins with a large shock on the upper surface and every tested indicator clusters large numbers of cross-sectional control points in this region. This is exactly what the adaptive algorithm is designed to do; however, excessive clustering can cause numerical issues with the B-splines that would not be captured

Table 4 Results summary for CRM case. Unconverged cases denoted with $*$.

Case	Design Iterations (Convergence)	Max DVs	Drag (Counts)
S0	50	54	203.5
S1	150	165	201.2
S2	300	567	200.5
S3*	728	2091	201.1
S4*	420	8019	201.3
P	306	8019	199.9
$\mathcal{A}_{\text{QB}}$*	296	1875	200.1
$\mathcal{A}_{\text{QR}}$*	273	1469	200.7
$\mathcal{A}_{\text{QIS}}$	250	1999	199.8
$\mathcal{A}_{\text{QI}}$	200	2380	200.0

(a) Region 1 **(b) Region 2**

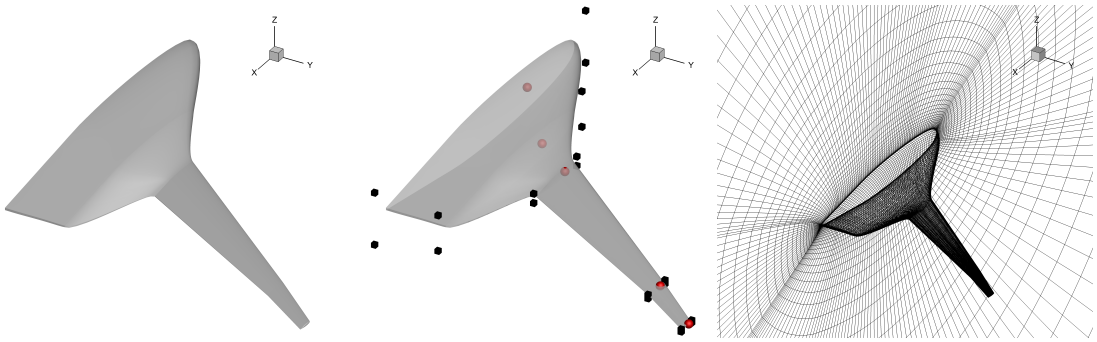
Fig. 16 Chordwise clustering of control points in two regions for $\mathcal{A}_{\text{QIS}}$ (Green) and $\mathcal{A}_{\text{QI}}$ (Orange)

by the potential indicator, leading to poor convergence and candidates underperforming their potential. Figure 16 compares the chordwise distribution of cross-sectional control points for the $\mathcal{A}_{\text{QI}}$ and $\mathcal{A}_{\text{QIS}}$ indicators at two key points along the surface of the wing after 100 function evaluations, and it is clear that the former is substantially less clustered simply by virtue of having fewer control points. This stands in contrast to the $\mathcal{A}_{\text{QB}}$ indicator, which appears to have been hampered by having too few design variables. The difference likely lies in the timing: $\mathcal{A}_{\text{QI}}$ benefited early on by focusing control around the shock without over-clustering; $\mathcal{A}_{\text{QB}}$ and $\mathcal{A}_{\text{QI}}$ suffered, relatively speaking, early on due to their degree of clustering. Once the shock had been largely eliminated and further global refinement was needed to continue optimization, the more conservative $\mathcal{A}_{\text{QB}}$ could not match the performance of the more aggressive $\mathcal{A}_{\text{QIS}}$ indicator.

Together, these results suggest several conclusions: 1) adaptive and progressive geometry control are capable of offering improved performance and increased automation in detailed design; 2) adaptive performance can be further improved by developing a method to prevent excessive clustering while still permitting the algorithm to focus control, and 3) the criteria for terminating adaptive refinement may require further tweaks to ensure performance is consistent across different indicators. The latter two points in particular are a focus of ongoing development work on the DGC algorithm.

C. Transonic, Viscous Hybrid Wing-Body Optimization

The final examined case is the transonic lift-constrained drag-minimization of an HWB subject to the RANS equations. The active degrees of freedom are twist, taper, and section shape, in addition to an angle of attack design variable. Enforced non-linear constraints include pitching moment, minimum volume, and a cabin shape constraint meant to ensure adequate passenger capacity in the final geometry. The problem can be formally stated as



(a) Baseline geometry **(b) Baseline (S0) geometry control** **(c) Baseline mesh**

Fig. 17 Geometry, mesh and control for HWB baseline design

Table 5 HWB mesh parameters

Nodes	2,314,368
$(\text{Nodes})^{-\frac{2}{3}}$	5.72×10^{-5}
Blocks	128
Off-Wall Spacing	3.5×10^{-7} MAC
y^+	0.8

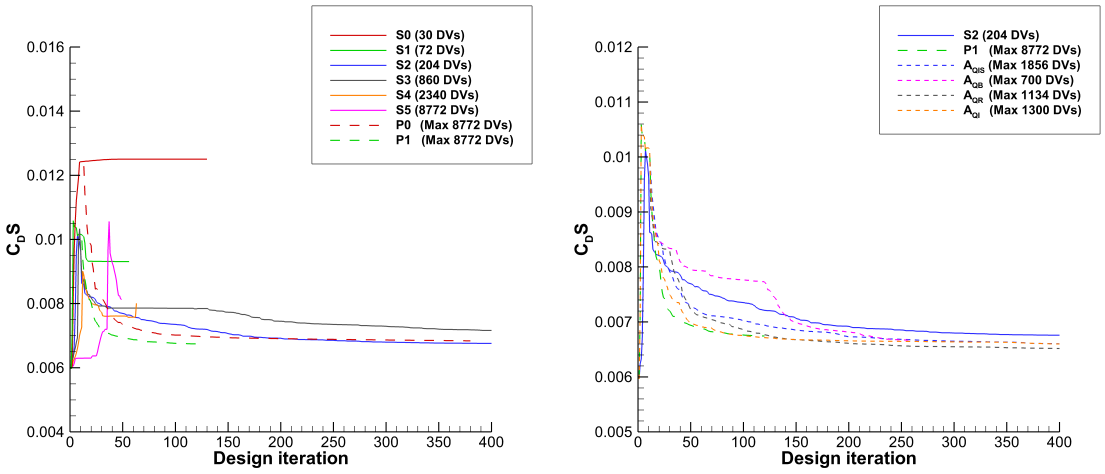
$$
\begin{aligned}
\text{minimize} \quad & C_D S \\
\text{w.r.t.} \quad & \mathbf{X} \\
\text{subject to} \quad & C_L S = 0.12\,\text{MAC}^2 \\
& V \geq 0.0786\,\text{MAC}^3 \\
& C_M S = 0
\end{aligned}
\tag{18}
$$

where S is the projected area, and optimization is performed at a Mach number of 0.78 and a Reynolds number of 76 million. This is representative of preliminary design problems often encountered in the literature [11, 12], characterized by the ability of the optimizer to create larger changes to the design, particularly planform changes, but with less freedom than a fully-exploratory problem. The baseline geometry, geometry control, and mesh are shown in Figure 17, with optimization mesh properties tabulated in Table 5. The baseline geometry control in Figure 17b is much coarser than would generally be used for a problem such as this: there are just four control points per cross-section, arranged in two rows of two, and there are only enough cross-sections to define the critical geometric transition points in the geometry. The axial control points are similarly coarse, but as these are not used for this problem they are shown for illustration purposes only. The geometry control was designed this way intentionally to establish a floor on the performance of DGC performed by an inexperienced user initialized with an unfit-for-purpose baseline geometry control.

In the same vein as the CRM study, this baseline geometry control scheme was uniformly enriched by inserting a new control point between each existing pair in order to create a sequence of static schemes labelled S0 through S5. To represent an inexperienced practitioner, all of the static schemes use 2$^{\text{nd}}$ order ($p = 1$) B-splines, and to maintain a fair comparison the DGC algorithms are not permitted to increase their order beyond 2. Due to the coarseness of the S0 level, six static schemes were created rather than the five used previously, and all six static levels were then optimized as far as possible. Alongside the static cases, the progressive and adaptive DGC algorithms were also applied, using the parameters given in Table 6. Due to the coarseness of the S0 geometry control we deviate slightly from the CRM test procedure here. It was expected that S0 would struggle and S1, while still coarse, represents a more reasonable initial geometry control; therefore two versions of the progressive geometry control were optimized: P0 which begins in S0 and moves through all six static schemes, and P1 which begins in S1 and moves through the last five of the static schemes. This allows P0 to test DGC performance starting from a manifestly unsuitable geometry control scheme and

Table 6 DGC parameters for HWB case Modify if parameters change

Parameter	Value
r_{loc}	0.010
r_{glb}	0.001
n_{loc}	5
n_{glb}	25
a_{loc}	0.100
a_{glb}	0.005
m_{loc}	2
m_{glb}	3



(a) Static and progressive **(b) P1 and adaptive**

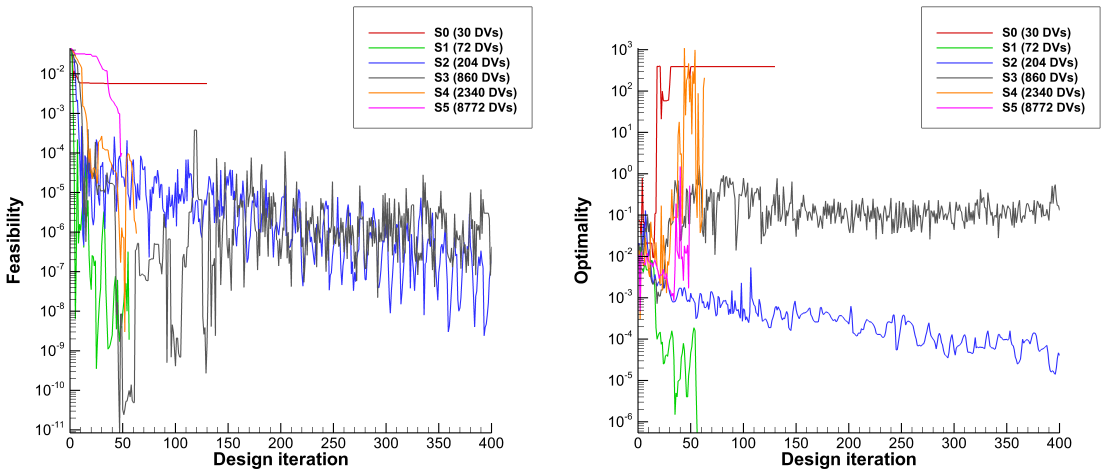
Fig. 18 Drag histories for HWB case

P1 to test performance from a more sensible starting point. All of the adaptive DGC methods were initialized in S1. Drag convergence for all cases is plotted in Figure 18; to more clearly show the differing performance between the two progressive cases we have provided two plots, Figure 18a compares the static cases with both progressive approaches, while Figure 18b compares the four adaptive tested adaptive methods with the best progressive and static approaches. Finally, overall results are summarized in Table 7.

As expected, S0 fails to meaningfully make progress; it is unable to reduce drag and lacks the necessary control resolution to satisfy the optimization constraints, causing the optimizer to stall with extremely large values for optimality and feasibility. Even for the static schemes which converged, insufficient control has a stark effect on final drag, with S1 converging to a final drag 50% larger than S2. While too few design variables is clearly a potential problem in this case, as we refine past S2 we observe that too many design variables is equally problematic. S3 runs for over 800 design iterations before ultimately failing while S4 and S5 stall and exit within the first 100 iterations.

Figure 18a plots the drag convergence for both progressive cases alongside the static methods. Both progressive approaches perform well, and Table 7 indicates that the final drag P0 locates is approximately 1% higher than the best static case, but it converges to it roughly 20% faster. Both cases fully converged, so the difference in drag may represent a degree of multimodality. P1, starting from the more reasonable S1 geometry control, produces nearly ideal performance, converging 70% faster than the best static case to a slightly lower final drag.

Adaptive convergence is plotted in Figure 18b compared to the best progressive and static schemes, and we note that all of the adaptive cases outperform the static geometry control schemes, offering faster and deeper convergence. Looking broadly at the adaptive results, while all outperform the static cases and some outperform the progressive



(a) Feasibility **(b) Optimality**

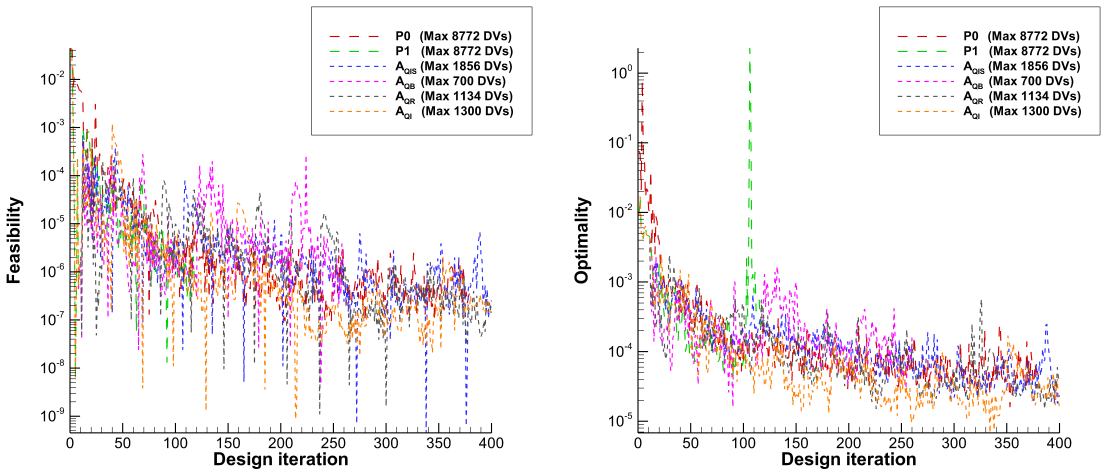
Fig. 19 Static convergence for HWB case

Table 7 Results summary for HWB case. Unconverged cases denoted with *.

Case	Design Iterations (Convergence)	Max DVs	$C_D S \left(\text{MAC}^2\right)$
S0*	130	30	1.25×10^{-2}
S1	25	72	9.31×10^{-3}
S2	350	204	6.75×10^{-3}
S3*	480	860	6.80×10^{-3}
S4*	63	2340	8.01×10^{-3}
S5*	49	8772	8.13×10^{-3}
P_0	275	8772	6.84×10^{-3}
P_1	100	8772	6.74×10^{-3}
$\mathcal{A}_{\text{QB}}$	257	700	6.66×10^{-3}
$\mathcal{A}_{\text{QR}}$	275	1134	6.51×10^{-3}
$\mathcal{A}_{\text{QIS}}$	350	1856	6.60×10^{-3}
$\mathcal{A}_{\text{QI}}$	250	1300	6.63×10^{-3}

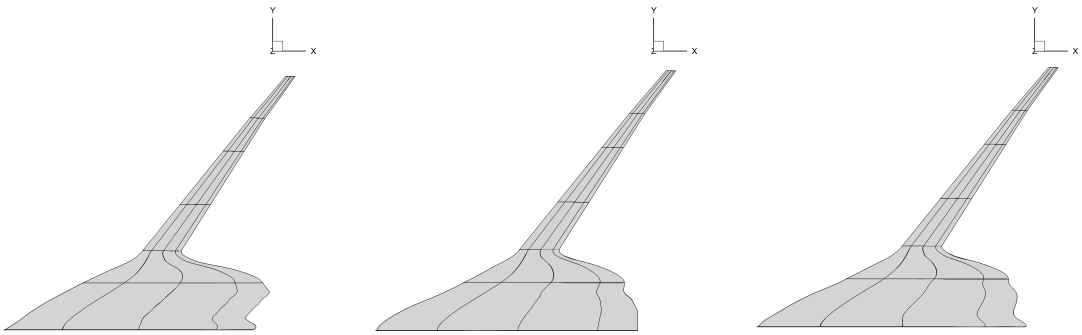
algorithm, there is a nearly 2% range in final drag and a 30% range in convergence time. Examining the final planform shapes and geometry control schemes for three of the best performing DGC cases, shown in Figures 21 and 22, reveals that these differences in performance may be partially due to multimodality.

The final geometries in Figure 21 are clearly distinct from one another; this is most obvious when comparing the P1 final geometry to the adaptive results, but even the geometric differences between the $\mathcal{A}_{\text{QI}}$ and $\mathcal{A}_{\text{QR}}$ results, while subtle, are sufficient to produce a 1.4% difference in final drag (equivalent to approximately 1.6 drag counts). These differences would be enough to qualify each of these three results as a distinct local optimum according to the criteria in [26], and that same work also confirms the presence of multimodality in design spaces similar to this. The concept of multimodality is somewhat thorny in the context of DGC, as all three of these cases are technically independent design spaces. However, we would expect that with sufficient refinement similar optimal geometries would be contained within all three. Given the very fine nature of the final geometry control schemes shown in Figure 22, we believe that this



(a) Feasibility **(b) Optimality**

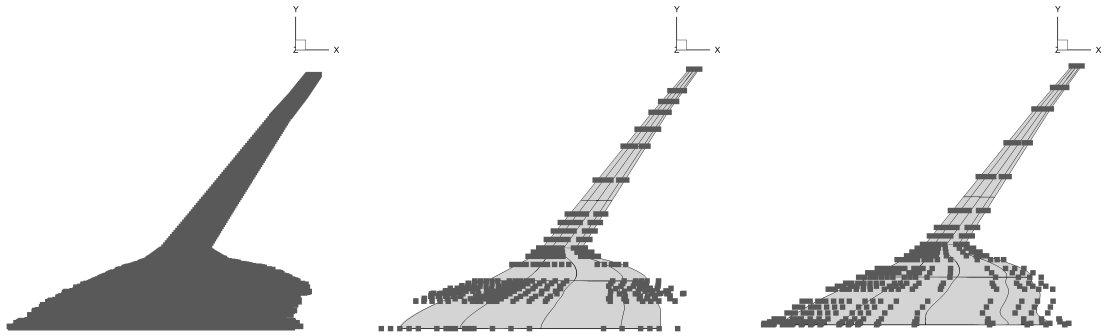
Fig. 20 DGC convergence for HWB case



(a) P1 **(b) $\mathcal{A}_{\mathbf{QI}}$** **(c) $\mathcal{A}_{\mathbf{QR}}$**

Fig. 21 Final geometries for selected DGC cases

should be the case. Therefore we are left with the conclusion that our DGC cases may be converging to distinct local optima; this complicates our assessment of their performance, as it is difficult to separate which gains or losses are due to the quality of the algorithm and which are due to the quality of the local optimum the algorithm converged to. A study of the relationship between DGC and multimodality to determine, among other questions, whether adaptive geometry control has any effect on the amount of multimodal risk in a design space is an obvious candidate for future investigation. Despite this ambiguity, we reiterate that all of the adaptive cases perform well and provide reliable convergence with a high degree of automation.



(a) P1 **(b) $\mathcal{A}_{\mathbf{QI}}$** **(c) $\mathcal{A}_{\mathbf{QR}}$**

Fig. 22 Final geometry control for selected DGC cases

V. Conclusions

A novel dynamic geometry control algorithm has been developed and validated before being applied to two aerodynamic shape optimization problems typical of detailed and preliminary design, respectively. Both progressive and adaptive algorithms have proven capable of improving convergence while increasing automation, with the adaptive geometry control generally offering deeper and occasionally faster convergence relative to the progressive approach. The benefits of DGC are particularly pronounced in higher-dimensional preliminary design problems but gains are also apparent in lower-dimensional detailed design problems. Results confirm the value of a quadratic fit for estimating the potential of each candidate search space, but also demonstrate that accurate Hessian data is of lesser importance. A quadratic fit based on a BFGS approximation of the Hessian performs poorly compared to simpler approximations, while one based on the SR1 Hessian approximation produces variable results, achieving among the best results in one case and the worst in another. Key areas requiring further study are preventing over-clustering in adaptive DGC, improvements to the adaptive refinement termination criterion, and the intersection of multimodality and dynamic geometry control. The latter could be effectively studied by combining the developed DGC algorithms with the gradient-based multistart (GBMS) approach of Chernukhin and Zingg [68].

Acknowledgements

The authors wish to gratefully acknowledge the financial support of Bombardier Aerospace and of the Government of Ontario through the Ontario Graduate Scholarship. All cases were performed using computational resources generously provided by Compute Canada.

References

[1] Jameson, A., "Aerodynamic design via control theory," *Journal of Scientific Computing*, Vol. 3, No. 3, 1988, pp. 233–260, doi:https://doi.org/10.1007/BF01061285.

[2] Jameson, A., Martinelli, L., and Pierce, N. A., "Optimum aerodynamic design using the Navier–Stokes equations," *Theoretical and Computational Fluid Dynamics*, Vol. 10, No. 1, 1998, pp. 213–237, doi:https://doi.org/10.1007/s001620050060.

[3] Bisson, F. and Nadarajah, S., "Adjoint-based aerodynamic optimization of benchmark problems," *53rd AIAA Aerospace Sciences Meeting*, No. 2015-1948, 2015, doi:https://doi.org/10.2514/6.2015-1948.

[4] Méheut, M., Destarac, D., Ben Khelil, S., Carrier, G., Dumont, A., and Peter, J., "Gradient-based single and multi-points aerodynamic optimizations with the elsA software," *53rd AIAA Aerospace Sciences Meeting*, No. 2015-0263, 2015, doi:https://doi.org/10.2514/6.2015-0263.

[5] Lyu, Z. and Martins, J., "Aerodynamic design optimization studies of a blended-wing-body aircraft," *Journal of Aircraft*, Vol. 51, No. 5, 2014, pp. 1604–1617, doi:https://doi.org/10.2514/1.C032491.

[6] Mura, G. L., Hinchliffe, B. L., Qin, N., and Brezillon, J., "Nonconsistent mesh movement and sensitivity calculation on adjoint aerodynamic optimization," *AIAA Journal*, Vol. 56, No. 4, 2017, pp. 1541–1553, doi:https://doi.org/10.2514/1.J055904.

[7] Epstein, B., Jameson, A., Peigin, S., Roman, D., Harrison, N., and Vassberg, J., "Comparative study of three-dimensional wing drag minimization by different optimization techniques," *Journal of Aircraft*, Vol. 46, No. 2, 2009, pp. 526–541, doi:https://doi.org/10.2514/1.38216.

[8] Reist, T. A., Koo, D., Zingg, D. W., Bochud, P., Castonguay, P., and Leblond, D., "Cross Validation of Aerodynamic Shape Optimization Methodologies for Aircraft Wing-Body Optimization," *AIAA Journal*, 2020, pp. 1–15, doi:https://doi.org/10.2514/1.J059091.

[9] Chen, S., Lyu, Z., Kenway, G., and Martins, J., "Aerodynamic shape optimization of common research model wing–body–tail configuration," *Journal of Aircraft*, Vol. 53, No. 1, 2016, pp. 276–293, doi:https://doi.org/10.2514/1.C033328.

[10] Ronzheimer, A., Hepperle, M., Brezillon, J., Brodersen, O., and Lieser, J., "Aerodynamic optimal engine integration at the fuselage tail of a generic business jet configuration," *New Results in Numerical and Experimental Fluid Mechanics VIII*, Springer, 2013, pp. 25–32, doi:https://doi.org/10.1007/978-3-642-35680-3_4.

[11] Reist, T. A., Zingg, D. W., Rakowitz, M., Potter, G., and Banerjee, S., "Multifidelity Optimization of Hybrid Wing–Body Aircraft with Stability and Control Requirements," *Journal of Aircraft*, Vol. 56, No. 2, 2019, pp. 442–456, doi:https://doi.org/10.2514/1.C034703.

[12] Meheut, M., Arntz, A., and Carrier, G., "Aerodynamic shape optimizations of a blended wing body configuration for several wing planforms," *30th AIAA Applied Aerodynamics Conference*, No. 2012-3122, 2012, doi:https://doi.org/10.2514/6.2012-3122.

[13] Reist, T. A. and Zingg, D. W., "High-fidelity aerodynamic shape optimization of a lifting-fuselage concept for regional aircraft," *Journal of Aircraft*, Vol. 54, No. 3, 2017, pp. 1085–1097, doi:https://doi.org/10.2514/1.C033798.

[14] Gagnon, H. and Zingg, D. W., "Aerodynamic optimization trade study of a box-wing aircraft configuration," *Journal of Aircraft*, Vol. 53, No. 4, 2016, pp. 971–981, doi:https://doi.org/10.2514/1.C033592.

[15] Secco, N. and Martins, J., "RANS-based aerodynamic shape optimization of a strut-braced wing with overset meshes," *Journal of Aircraft*, Vol. 56, No. 1, 2019, pp. 217–227.

[16] Chau, T. and Zingg, D. W., "Aerodynamic shape optimization of a box-wing regional aircraft based on the reynolds-averaged Navier-Stokes equations," *35th AIAA Applied Aerodynamics Conference*, No. 2017-3258, 2017, doi:https://doi.org/10.2514/6.2017-3258.

[17] "Consolidated statement of continuing ICAO policies and practices related to environmental protection - Climate change," 2019.

[18] Fraga, J., "Boeing 787: From the Ground Up," *Aero Magazine*, Vol. 4, 2006.

[19] Nathan, S., "The wing master: Bombardier's award-winning aerodynamic production," *The Engineer*, October 2019.

[20] Owens, R., Hasel, K., and Mapes, D., "Ultra high bypass turbofan technologies for the twenty-first century," *26th Joint Propulsion Conference*, 1990, p. 2397, doi:https://arc.aiaa.org/doi/abs/10.2514/6.1990-2397.

[21] Shuba, E. S. and Kifle, D., "Microalgae to biofuels:'Promising'alternative and renewable energy, review," *Renewable and Sustainable Energy Reviews*, Vol. 81, 2018, pp. 743–755, doi:https://doi.org/10.1016/j.rser.2017.08.042.

[22] Piotrowski, M. and Zingg, D., "Smooth Local Correlation-Based Transition Model for the Spalart–Allmaras Turbulence Model," *AIAA Journal*, Vol. 59, No. 2, 2021, pp. 474–492.

[23] Hicken, J. E. and Zingg, D. W., "Parallel Newton-Krylov solver for the Euler equations discretized using simultaneous approximation terms," *AIAA Journal*, Vol. 46, No. 11, 2008, pp. 2773–2786, doi:https://doi.org/10.2514/1.34810.

[24] Koo, D. and Zingg, D. W., "Investigation into Aerodynamic Shape Optimization of Planar and Nonplanar Wings," *AIAA Journal*, Vol. 56, No. 1, 2018, pp. 1–14, doi:https://doi.org/10.2514/1.j055978.

[25] Wang, Q., Hu, R., and Blonigan, P., "Least squares shadowing sensitivity analysis of chaotic limit cycle oscillations," *Journal of Computational Physics*, Vol. 267, 2014, pp. 210–224.

[26] Streuber, G. M. and Zingg, D. W., "Evaluating the Risk of Local Optima in Aerodynamic Shape Optimization," *AIAA Journal*, Vol. 59, No. 1, 2021, pp. 75–87.

[27] Fudge, D., Zingg, D., and Haimes, R., "A CAD-free and a CAD-based geometry control system for aerodynamic shape optimization," *43rd AIAA Aerospace Sciences Meeting and Exhibit*, No. 2005-451, 2005.

[28] Kenway, G., Kennedy, G., and Martins, J., "A CAD-free approach to high-fidelity aerostructural optimization," *13th AIAA/ISSMO Multidisciplinary Analysis Optimization Conference*, No. 2010-9231, 2010.

[29] Vučina, D., Marinić-Kragić, I., and Milas, Z., "Numerical models for robust shape optimization of wind turbine blades," *Renewable Energy*, Vol. 87, 2016, pp. 849–862.

[30] Hicks, R. M. and Henne, P. A., "Wing design by numerical optimization," *Journal of Aircraft*, Vol. 15, No. 7, 1978, pp. 407–412.

[31] Secco, N., Jasa, J., Kenway, G., and Martins, J., "Component-based geometry manipulation for aerodynamic shape optimization with overset meshes," *AIAA Journal*, Vol. 56, No. 9, 2018, pp. 3667–3679.

[32] Jakobsson, S. and Amoignon, O., "Mesh Deformation using Radial Basis Functions for Gradient-Based Aerodynamic Shape Optimization," *Computers and Fluids*, Vol. 36, 2007, pp. 1119–1136.

[33] da Silva, J. P., Giraldi, G. A., and Apolinário Jr, A. L., "Data-driven optimization approach for mass-spring models parametrization based on isogeometric analysis," *Journal of Computational Science*, Vol. 23, 2017, pp. 1–19.

[34] Lu, X., Huang, J., Song, L., and Li, J., "An improved geometric parameter airfoil parameterization method," *Aerospace Science and Technology*, Vol. 78, 2018, pp. 241–247.

[35] Limkilde, A., Evgrafov, A., Gravesen, J., and Mantzaflaris, A., "Practical isogeometric shape optimization: parametrization by means of regularization," *Journal of Computational Design and Engineering*, 2020.

[36] Agromayor, R., Anand, N., Müller, J.-D., Pini, M., and Nord, L., "A Unified Geometry Parametrization Method for Turbomachinery Blades," *Computer-Aided Design*, Vol. 133, 2021, pp. 102987.

[37] Cinquegrana, D. and Iuliano, E., "Efficient Global Optimization Method for Multipoint Airfoil Design," *Advances in Evolutionary and Deterministic Methods for Design, Optimization and Control in Engineering and Sciences*, Springer, 2019, pp. 95–114, doi:https://doi.org/10.1007/978-3-319-89988-6_6.

[38] Chen, W. and Ramamurthy, A., "Deep Generative Model for Efficient 3D Airfoil Parameterization and Generation," *AIAA Scitech 2021 Forum*, No. 2021-1690, 2021.

[39] Kedward, L., Allen, C. B., and Rendall, T., "Gradient-Limiting Shape Control for Efficient Aerodynamic Optimization," *AIAA Journal*, Vol. 58, No. 9, 2020, pp. 3748–3764.

[40] Beux, F. and Dervieux, A., "A Hierarchical approach for shape optimisation, Inria report 1868," *Attouch & Cominetti (1996)] Attouch H., Cominetti R*, 1996, pp. 519–540.

[41] Désidéri, J., "Hierarchical optimum-shape algorithms using embedded Bézier parameterizations," *Numerical Methods for Scientific Computing, Variational Problems and Applications*, 2003, pp. 45–56.

[42] Andreoli, M., Ales, J., and Désidéri, J.-A., "Free-form-deformation parameterization for multilevel 3D shape optimization in aerodynamics," Tech. Rep. 5019, INRIA, 2003.

[43] Duvigneau, R., Chaigne, B., and Désidéri, J.-A., "Multi-level parameterization for shape optimization in aerodynamics and electromagnetics using a particle swarm optimization algorithm," Tech. Rep. 6003, INRIA.

[44] Anderson, G. R., Nemec, M., and Aftosmis, M. J., "Aerodynamic shape optimization benchmarks with error control and automatic parameterization," *53rd AIAA Aerospace Sciences Meeting*, No. 2015-1719, 2015.

[45] Masters, D., Taylor, N., Rendall, T., and Allen, C., "Progressive subdivision curves for aerodynamic shape optimisation," *54th AIAA Aerospace Sciences Meeting*, No. 2016-0559, 2016.

[46] Masters, D., Taylor, N., Rendall, T., and Allen, C., "Multilevel subdivision parameterization scheme for aerodynamic shape optimization," *AIAA Journal*, Vol. 55, No. 10, 2017, pp. 3288–3303.

[47] Han, X. and Zingg, D. W., "An adaptive geometry parametrization for aerodynamic shape optimization," *Optimization and Engineering*, Vol. 15, No. 1, 2014, pp. 69–91.

[48] Masters, D. A., Taylor, N. J., Rendall, T., and Allen, C. B., "A Locally Adaptive Subdivision Parameterisation Scheme for Aerodynamic Shape Optimisation," *34th AIAA Applied Aerodynamics Conference*, No. 2016-3866, 2016.

[49] He, X., Li, J., Mader, C. A., Yildirim, A., and Martins, J., "Robust aerodynamic shape optimization - from a circle to an airfoil," *Aerospace Science and Technology*, Vol. 87, 2019, pp. 48–61.

[50] Anderson, G. R. and Aftosmis, M. J., "Adaptive Shape Control for Aerodynamic Design," *56th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, No. 2015-398, 2015.

[51] Anderson, G. R., *Shape Optimization in Adaptive Search Spaces*, Ph.D. thesis, Stanford University, 2016.

[52] Sinsay, J. and Alonso, J., "A Heuristic Approach to Finding the Preferred Design Variable Parameterization for Optimization," *57th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, No. 2016-0415, 2016.

[53] Osusky, M. and Zingg, D. W., "Parallel Newton–Krylov–Schur Flow Solver for the Navier–Stokes Equations," *AIAA Journal*, Vol. 51, No. 12, 2013, pp. 2833–2851, doi:https://doi.org/10.2514/1.j052487.

[54] Hicken, J. E. and Zingg, D. W., "Aerodynamic optimization algorithm with integrated geometry parameterization and mesh movement," *AIAA Journal*, Vol. 48, No. 2, 2010, pp. 400–413, doi:https://doi.org/10.2514/1.44033.

[55] Osusky, L., Buckley, H., Reist, T., and Zingg, D. W., "Drag minimization based on the Navier–Stokes equations using a Newton–Krylov approach," *AIAA Journal*, Vol. 53, No. 6, 2015, pp. 1555–1577.

[56] De Sturler, E., "Truncation strategies for optimal Krylov subspace methods," *SIAM Journal on Numerical Analysis*, Vol. 36, No. 3, 1999, pp. 864–889, doi:https://doi.org/10.1137/s0036142997315950.

[57] Gill, P. E., Murray, W., and Saunders, M. A., "SNOPT: An SQP algorithm for large-scale constrained optimization," *SIAM Review*, Vol. 47, No. 1, 2005, pp. 99–131, doi:https://doi.org/10.1137/s1052623499350013.

[58] Broyden, C. G., "The convergence of a class of double-rank minimization algorithms 1. general considerations," *IMA Journal of Applied Mathematics*, Vol. 6, No. 1, 1970, pp. 76–90.

[59] Fletcher, R., "A new approach to variable metric algorithms," *The Computer Journal*, Vol. 13, No. 3, 1970, pp. 317–322.

[60] Goldfarb, D., "A family of variable-metric methods derived by variational means," *Mathematics of Computation*, Vol. 24, No. 109, 1970, pp. 23–26.

[61] Shanno, D., "Conditioning of quasi-Newton methods for function minimization," *Mathematics of Computation*, Vol. 24, No. 111, 1970, pp. 647–656.

[62] Truong, A. H., Oldfield, C. A., and Zingg, D. W., "Mesh movement for a discrete-adjoint Newton-Krylov algorithm for aerodynamic optimization," *AIAA Journal*, Vol. 46, No. 7, 2008, pp. 1695–1704, doi:https://doi.org/10.2514/1.33836.

[63] Sederberg, T. W. and Parry, S. R., "Free-form deformation of solid geometric models," *Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques*, 1986, pp. 151–160.

[64] Samareh, J. A., "Aerodynamic shape optimization based on free-form deformation," *10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, No. 2004-4630, 2004, pp. 3672–3683.

[65] Squire, W. and Trapp, G., "Using complex variables to estimate derivatives of real functions," *SIAM Review*, Vol. 40, No. 1, 1998, pp. 110–112.

[66] Boehm, W., "Inserting new knots into B-spline curves," *Computer-Aided Design*, Vol. 12, No. 4, 1980, pp. 199–201.

[67] Nocedal, J. and Wright, S., *Numerical Optimization*, Vol. 35, Springer Science, 1999.

[68] Chernukhin, O. and Zingg, D. W., "Multimodality and Global Optimization in Aerodynamic Design," *AIAA Journal*, Vol. 51, No. 6, 2013, pp. 25–34, doi:https://doi.org/10.2514/1.j051835.