AIAA 2010-9231

13th AIAA/ISSMO Multidisciplinary Analysis Optimization Conference
September 13–15, 2010, Fort Worth, Texas, United States

A CAD-Free Approach to High-Fidelity Aerostructural Optimization

Gaetan K.W. Kenway,*
Graeme J. Kennedy, †
University of Toronto Institute for Aerospace Studies, Toronto, ON, Canada
Joaquim R. R. A. Martins‡
Department of Aerospace Engineering, University of Michigan, Ann Arbor, MI

Geometry parametrization for high-fidelity multidisciplinary optimization is an important and complex problem. We present a CAD-free geometry parametrization method using a free-from deformation volume approach. This approach yields several important advantages over other parametrization techniques, the most of important of which is the efficient computation of analytic derivatives for gradient-based optimization. A parallel, hybrid, algebraic-linear-elasticity mesh perturbation scheme which produces high quality perturbed meshes with low computational effort is also presented. We couple an Euler CFD solver with a finite-element model that uses fourth-order degenerate shell elements. As a demonstration problem, we perform the aerostructural redesign of a subsonic wing for transonic flight conditions. We show that this optimization problem captures some of the complex multidisciplinary trade-offs inherent in wing design.

I. Introduction

In 1965, Gorden E. Moore, an engineer at Fairchild Semiconductors, observed that the number of components on an integrated circuit approximately doubled every 18 months.[1] Numerous technological improvements have allowed the semiconductor industry to keep pace with what has become known as "Moore's Law". The effect of over four decades of phenomenal computational improvements has had a profound effect on the aerospace industry. Engineers can now perform complex physics-based simulations on a desktop computer that would have required a parallel computing cluster only a decade ago. This continual increase in computing power has enabled engineers to perform much of the design and analysis of complex systems without the use of physical scale models and prototypes. Computational methods allow engineers to validate designs much earlier in the design process, potentially saving costly redesigns later in the design stage.

This paper focuses on the handling of geometry and meshes used in these simulations. Even with today's parallel computational resources, optimization with respect to large numbers of design variables employing high-fidelity analysis requires the use of of gradient-based optimization with efficient sensitivity analysis techniques. CAD-driven approaches work, but a continuous parameterization of the geometry and the analysis is necessary for design optimization. This is difficult to achieve with CAD. An even more important factor is the cost of obtaining sensitivity information from CAD, which at the moment is too costly, especially for large numbers of design variables. Our simplified approach eliminates CAD from the optimization leading to a parametrization well suited to multidisciplinary optimization (MDO).

The use of Computational Fluid Dynamics (CFD) and Computational Structural Mechanics (CSM) has become widespread in the aerospace industry. New aircraft and engine designs are extensively validated using

*Ph.D. Candidate, AIAA Student Member
†Ph.D Candidate, AIAA Student Member
‡Associate Professor, AIAA Senior Member

these methods, which have contributed to the significant simultaneous increases in aerodynamic, structural and propulsive performance since the beginning of the jet age.[2,3] A natural extension to performing computational analysis is the use of numerical optimization. Three-dimensional shape optimization of aerodynamic surfaces using computational fluid dynamics have been thoroughly investigated and continues to be an area of active research.[4–7] The adjoint sensitivity method has permitted optimizations with hundreds of design variables with low computational cost compared to finite-difference approaches.[8]

It has been shown that repeated sequential optimization of individual disciplines does not necessarily result in an optimal multidisciplinary system.[9] This requirement to consider all disciplines involved simultaneously, resulted in the emergence of Multidisciplinary Design Optimization (MDO). Two factors have slowed industry's adoption of MDO. The first is the increased computational cost and complexity of the optimization problem. The second, and perhaps more important factor, is that the inter-disciplinary nature of MDO does not integrated easily into well established aerodynamics and structural design groups. There are however, instances where MDO has appeared in the conceptual and preliminary design stages.[10–12] Due to the highly coupled nature of the aerostructural problem, a Multidisciplinary Design Feasible architecture approach is employed casting the problem as a single set of coupled equations.[13,14] A nonlinear block Gauss–Sidel scheme with an Aitken acceleration technique is employed to converge this system.[15]

In this work we present the tools and techniques necessary to conduct high-fidelity MDO for a coupled two discipline problem involving aerodynamics and structural disciplines. The goal is to produce an optimization procedure suitable for use during the preliminary design stage. This paper first describes the CAD–free parameterization technique and a new hybrid mesh perturbation scheme. We then describe the discipline solver, the multidisciplinary coupling methodology and optimization algorithm. Finally, we present results of a model problem: the redesign of a subsonic transport wing for transonic conditions.

II. Parametrization for MDO

For multi-disciplinary optimization problems, the choice of parametrization is not trivial. We wish to use high fidelity tools, to allow an engineer or optimization algorithm to make better design decisions and trade-offs earlier in the design cycle. Unlike low fidelity methods, such as vortex-lattice models and equivalent beam models, CFD and CSM require detailed discretizations of the computational domain. These discretizations are inextricably linked to the accuracy of the solutions.

CFD analysis for MDO systems typically models the "wetted surface" or "outer mold line" (OML) of the structure. Moreover, the computational domain is three dimensional and may include millions or tens of millions of nodes. In contrast, the CSM models require not only the wetted surface but also a description of the internal aircraft structure including ribs, skins, spars, and stiffeners. Samareh[16] identifies eight types of geometric parametrizations: basis vector, domain element, partial differential equation, discrete, polynomial and spline, CAD-based analytical, and free-form deformation (FFD). We discuss our CAD-free parametrization approach which utilizes both the FFD volume based approach.

A. CAD–Free Architecture

We refer to the geometry tools we present in the following section as `pyPSG`—Python Parametric Surface Geometry. It includes functionality for working with both B-spline curves, surfaces and volumes.

1. `pySpline`

This is the underlying B-spline library for curves, surfaces and volumes used in `pyPSG`. The evaluation routines are written in Fortran90, and wrapped using `f2py`[17] to provide a high-level, object-oriented API in Python. The recurrence relations for the B-spline basis functions are

$$\mathcal{N}_{i,0}(u) = \begin{cases} 1 & \text{if } t_i \leq u \ < t_{i+1} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

$$\mathcal{N}_{i,p}(u) = \frac{u - t_i}{t_{i+p} - t_i}\mathcal{N}_{i,p-1} + \frac{t_{i+p+1} - u}{t_{i+p+1} - t_{i+1}}\mathcal{N}_{i+1,p-1}, \tag{2}$$

where u is the parametric location with respect to knots t_i. Parametric equations for B-spline curves, surfaces and volumes can be written as

$$C(u) = \sum_{i=0}^{\mathrm{N}_u-1} \mathcal{N}_{i,p_u}(u) P_i \tag{3}$$

$$S(u,v) = \sum_{i=0}^{\mathrm{N}_u-1} \sum_{j=0}^{\mathrm{N}_v-1} \mathcal{N}_{i,p_u}(u) \mathcal{N}_{j,p_v}(v) P_{i,j} \tag{4}$$

$$V(u,v,w) = \sum_{i=0}^{\mathrm{N}_u-1} \sum_{j=0}^{\mathrm{N}_v-1} \sum_{k=0}^{\mathrm{N}_w-1} \mathcal{N}_{i,p_u}(u) \mathcal{N}_{j,p_v}(v) \mathcal{N}_{k,p_w}(w) P_{i,j,k}, \tag{5}$$

where the control points, $\{P_{i,j,k}, i = 0, ..., \mathrm{N}_u - 1; j = 0, ..., \mathrm{N}_v - 1; k = 0, ..., \mathrm{N}_w - 1\}$, exist in 3 spatial dimensions. $\mathcal{N}_{i,p_u}$, $\mathcal{N}_{j,p_v}$ and $\mathcal{N}_{k,p_w}$ are the polynomial B-spline basis functions of degree p_u, p_v and p_w respectively. The polynomial functions are joined by strictly increasing vector sequences $T = t_0, ..., t_{\mathrm{N}+p+1}$ such that C^{p-1} continuity is ensured at each knot boundary. We choose to use open knot vectors of the form $T = \{0, 0, 0, 0, t_4, t_5, ..., t_{\mathrm{N\text{-}1}}, 1, 1, 1, 1\}$, where the number of repeated values correspond to one more than the spline degree $(p + 1)$. These $p + 1$ repeated knots ensure the spline passes exactly through the control points at the end of each edge. One of the key advantages of B-splines is the analytic formulation of the their derivatives. By differentiating the basis functions l times, we can evaluate the l^{th} order derivative with respect to the parameter value. The derivatives for curves, surfaces and volumes are

$$\frac{\partial^l}{\partial^l u} C(u) = \sum_{i=0}^{N_u-1} \mathcal{N}^{(l)}_{i,p_u} P_i \tag{6}$$

$$\frac{\partial^{l+m}}{\partial^l u \partial^m v} S(u,v) = \sum_{i=0}^{N_u-1} \sum_{j=0}^{N_v-1} \mathcal{N}^{(l)}_{i,p_u} \mathcal{N}^{(m)}_{j,p_v} P_{i,j} \tag{7}$$

$$\frac{\partial^{l+m+n}}{\partial^l u \partial^m v \partial^n w} V(u,v,w) = \sum_{i=0}^{N_u-1} \sum_{j=0}^{N_v-1} \sum_{k=0}^{N_w-1} \mathcal{N}^{(l)}_{i,p_u} \mathcal{N}^{(m)}_{j,p_v} \mathcal{N}^{(n)}_{k,p_w} P_{i,j,k}. \tag{8}$$

The first derivative of the B-spline basis function is

$$\mathcal{N}^{(1)}_{i,p} = \frac{p}{t_{i+p} - t_i} \mathcal{N}_{i,p-1}(u) - \frac{p}{t_{i+p+1} - t_{i+1}} \mathcal{N}_{i+1,p-1}(u). \tag{9}$$

The compact nature of the B-spline basis functions results in at most $p+1$ control points in a given direction will affecting a fixed parametric location. As a result of the linear nature of the B-spline shape functions, the derivative of a point in a volume at parametric location u, v, w, with respect to a control point $P_{i,j,k}$ is the product of shape functions

$$\frac{\partial V(u,v,w)}{\partial P_{i,j,k}} = \mathcal{N}_{i,p_u}(u) \mathcal{N}_{j,p_v}(v) \mathcal{N}_{k,p_w}(w). \tag{10}$$

B. Complex Configurations

Isolated curves, surfaces and volumes are rarely a complete description of a geometry of interest. It is generally necessary to combine the entities together in some topological manner. `pyNetwork`, `pyGeo`, and `pyBlock` are the Python modules that handle collections of curves, surfaces and volumes respectively. The topology of a given configuration is computed automatically using a robust algorithm that handles degenerate entities. We refer to topology as the connections between geometric entities and not the actual genus of the geometry. The topology calculation produces an abstract representation of geometry including a unique set of nodes, edges, faces and all sets of parallel edges.

Input to generate the `pyGeo` and `pyBlock` objects is typically from commercial meshing software. With a discrete representation of the geometry we perform a least-squares regression globally to obtain B-spline approximations of the surface or volume. C^0 continuity is ensured across boundaries, since shared edges possess identical control points and knot vectors by construction. Standard point inversion algorithms for curves, surfaces and volumes are included as well as curve-curve and curve-surface intersections. Figure 1 shows an example with B-spline surfaces (`pyGeo`) and B-spline volumes (`pyBlock`).

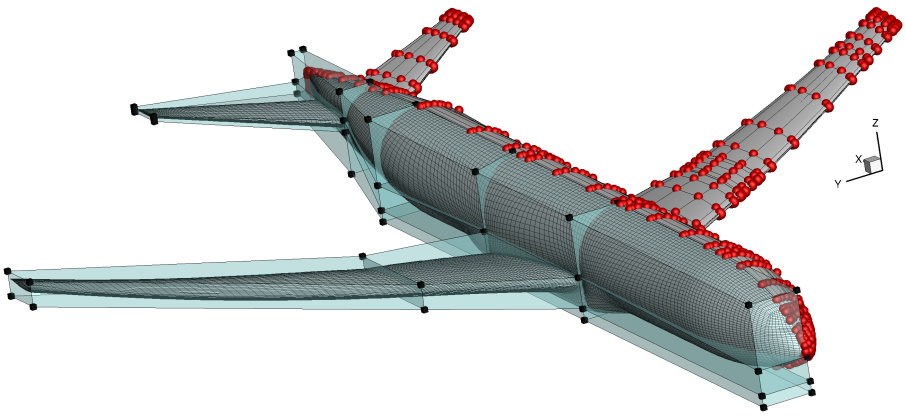
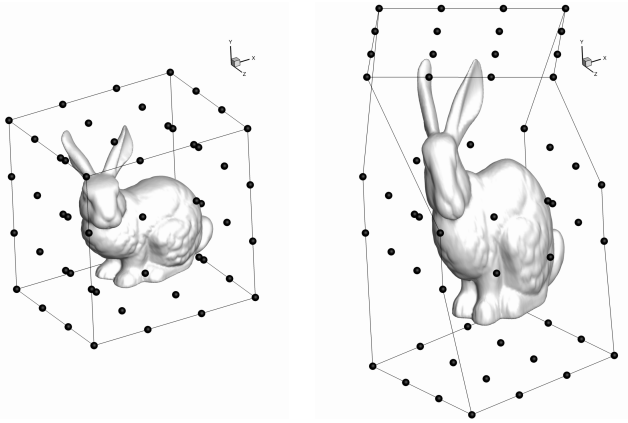


Figure 1: FFD deformation volumes (left) and tensor B-spline surface representation (right) of DPW4 geometry[18]

C. Free Form Deformation

The free-form deformation approach is borrowed from soft object animation in the computer graphics field.[19] The method can be most easily visualized as embedding an object in a clear, flexible, rubber-like material. This rubber is usually a relatively simple geometry and any $\mathcal{R}^3 \rightarrow \mathcal{R}^3$ mapping may be used. Typically, tri-variate Bézier, B-spline or NURBS volumes are utilized. Any geometry may be embedded inside the volume by performing a Newton search to determine the u, v, w values mapping parameter space to physical space. Once embedded, high level modifications made to the FFD lattice can be used to indirectly modify the embedded objects. Figure 2 shows the Stanford bunny undergoing twisting and stretching operations inside an FFD volume.[20]



(a) Stanford bunny embedded in a $4 \times 4 \times 4$ FFD lattice

(b) Modified bunny

Figure 2: Free-form deformation volume

1. *FFD for MDO*

The utility of free-form deformation volumes may not be immediately apparent. The key observation to make is that FFD volumes parametrize the *geometry change* rather than the geometry itself. It is therefore only necessary to use a set of design variables that span the desired modification rather than the geometry itself. This can lead to a more efficient and compact set of design variables. A brief description of the key aspects of utilizing FFD volumes for MDO are explained below.

Exact initial geometry: FFD volumes can represent initial geometry exactly (to numerical precision)

provided the inverse mapping search is tightly converged.

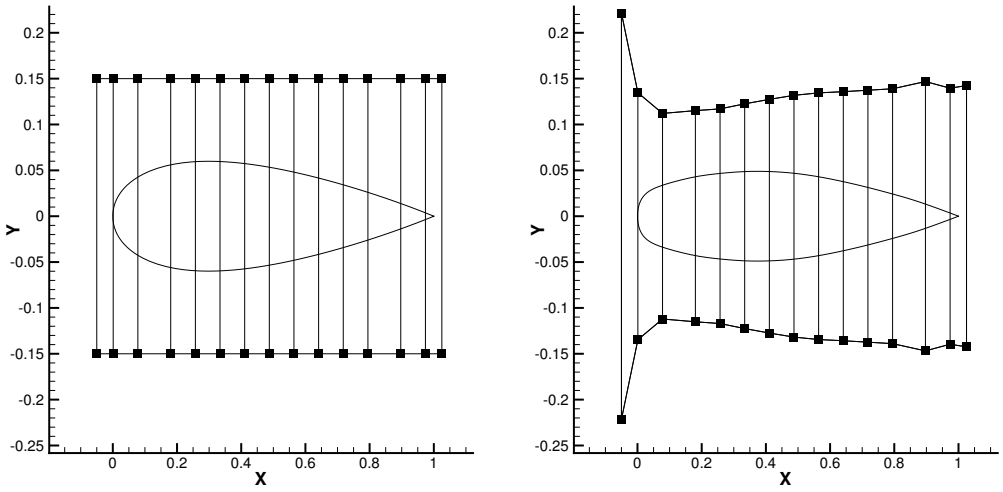
Geometric fidelity independent parametrization: Using a B-spline or CAD approach, the fidelity of the surface representation is tied to the resulting number of control points and hence the design variables. FFD allows simple sets of design variables to manipulate a complex geometry.

Constant topology: One of the key fundamental assertions we make for high-fidelity MDO is the topology of the geometry and the resulting discretizations for each discipline remain fixed. We require such a restriction to ensure the continuity of a function's gradient during the modification of design variables.

Reverse engineering: The FFD approach modifies point data only and thus it is possible modify existing non-parametric geometries. For example, if only a grid description of a geometry is available, a CAD-based approach would require a reverse engineering of the surfaces used to generated the original grid. This procedure may be time consuming and may not represent the original geometry exactly.

Global shape control: FFD volumes can be easily used for global shape control. Simply moving sets of control points together produces rigid-body motion perturbations. Additionally, a curve rigidly attached to the control points can facilitate these bulk motions.

Local shape control: It is also possible to use FFD volumes to perform smooth, high order shape perturbations to an arbitrary geometry. Figure 3 shows a NACA 0012 airfoil embedded inside an FFD surface (a 2D analog of a 3D FFD volume). A sequence of control point movements performs a smooth transformation into an ONERA M6 airfoil.



(a) NACA 0012 Airfoil

(b) ONERA M6 Airfoil

Figure 3: Local shape control modification required to transform NACA 0012 airfoil into an ONERA M6 airfoil

Analytic sensitivities: This is the most important advantage of the CAD-free approach to parametrization. Since the FFD volumes are simply tri-variate B-spline volumes we can write the sensitivity of any point inside the volume as

$$\frac{\partial X_{\text{pt}}}{\partial x_{\text{dv}}} = \frac{\partial X_{\text{coef}}}{\partial x_{\text{dv}}} \frac{\partial X_{\text{pt}}}{\partial X_{\text{coef}}} \tag{11}$$

where X_{pt} are the spatial coordinates of the embedded points, x_{dv} are the design variables and X_{coef} are the spatial coordinates of the FFD lattice control points. The $\partial X_{\text{pt}}/\partial X_{\text{coef}}$ partial consists of just the shape functions at the point's u, v, w coordinates, and the $\partial X_{\text{coef}}/\partial X_{\text{dv}}$ partial relates how the control points move with respect to the actual design variables. For local shape control, an individual control point may be a design variable, however, for global shape control many control points are generally grouped and moved together. `pyPSG` enables the use of user-supplied Python functions to

describe the design variable movement of control point groups. The derivatives of these functions are computed using the complex step-method.[21]

Consistent parametrization across disciplines: The modification of internal structural components poses no additional problem for FFD volumes. All internal structural components are embedded inside the volume with no additional special treatment.

2. `pyLayout`

`pyLayout` is the module used for automatic parametric structure generation. It automatically generates a finite-element model inside an outer-mold line given a description of the structural layout. This description includes the number and position of the ribs and spars, the thicknesses and material properties and ply angles for composite materials. The module then generates the structure and sets it directly in memory of our in-house solver, `pyTACS`.[22] The structure in Figure 11 is automatically generated using `pyLayout`.

III. Mesh Perturbation

The FFD volume approach works well for the parametrization of the structural domain as well as the wetted surface of the aircraft. However, unless the surface perturbations are of the same order as that of the grid off-wall spacing, at least a portion of the volume grid must be modified in order to prevent cells with negative volumes. The method we describe here is a combination of two well known approaches to mesh warping: algebraic methods and linear elasticity methods.

Linear elasticity methods are known to produce high quality perturbed meshes and can be applied to both structured and unstructured meshes. In the case of unstructured meshes, the linear spring analogy is widely used. The method replaces every mesh edge with a spring constant inversely proportional to its length. Dirichlet boundary conditions are imposed at both the moving boundary as well as the farfield. Originally developed for 2D-grids,[23] this technique alone can still produce tangled meshes and is usually augmented with a torsional spring analogy, which significantly improves its robustness, albeit at a higher computational cost. More recent applications to 3D have shown good performance.[24]

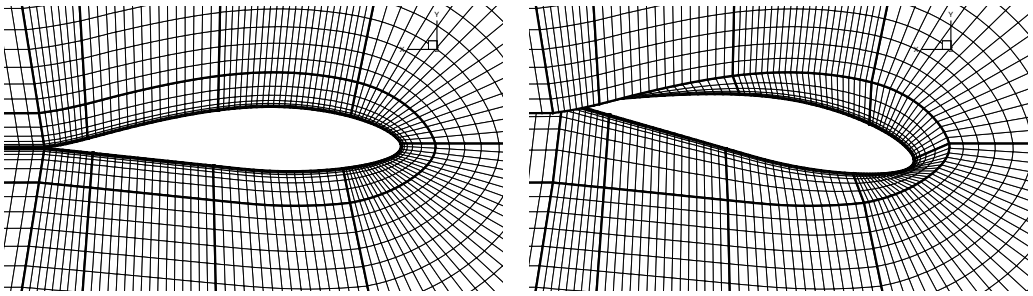
For structured grids, Troung et al.[25] use a similar approach. However, they use linear elastic elements corresponding to the cells in the CFD mesh. This method produces very high quality perturbed meshes, even for extreme displacements, however the cost of the algorithm is prohibitive for use in a multidisciplinary optimization setting. A modification of this scheme that uses the linear elastic warping acting on the control points of a cubic tensor B-spline volume grid has shown significant reduction in execution time while retaining many of the desirable characteristics of the linear elasticity approach.[26]

A. Algebraic Warping

We refer to the algebraic warping module as `pyWarp`.[27] In the simple case where only a single surface of a grid block is perturbed, the algorithm takes the following form:

$$X_V^{\text{new}} = X_V^{\text{old}} + \mathcal{S}^{\text{old}} \left(X_S^{\text{new}} - X_S^{\text{old}} \right), \tag{12}$$

where X_V are the volume grid coordinates, X_S are the surface grid points and $\mathcal{S}$ represents the arc length along the a given mesh line, normalized to 1. The formula is applied to each surface grid-line radiating from a displaced surface. However, in complex configurations, multiple faces may be perturbed by design variables and the scheme must be modified to resemble a transfinite interpolation (TFI). Unlike TFI, `pyWarp` uses the original distribution of interior points. While this method is very fast, it is not particularly robust. Problems begin to arise when the grid is forced to rotate and grid orthogonality is rapidly lost. Additionally, the algorithm can only attenuate perturbations within a single block of the surface. When small blocks are used near a surface, even very small perturbations result in a tangled mesh. This problem is illustrated in Figure 4. The rotation has produced inverted cells near the trailing edge and poor quality cells near the leading edge. In this case, the problem can be alleviated by allowing the blocks to extend to the far-field boundary. However, for more complex configurations, this option is not always possible.



(a) Original grid (b) 10°of twist

Figure 4: Algebraic warping scheme

B. Hybrid Warping Scheme

The idea behind the hybrid warping scheme is to apply a linear elasticity-based warping scheme to a coarse approximation of the mesh to account for large, low-frequency perturbations and to use the algebraic warping approach to attenuate small, high-frequency perturbations. The goal is to compute a similar, high quality perturbed mesh as obtained by using a linear elasticity scheme but at much lower computational cost. The basic outline of the algorithm is as follows:.

1. Select a subset of m nodes from each edge of N nodes. These nodes will form a coarse "super mesh". See Figure 5(b).
2. Apply linear-elasticity equations to the "super mesh". See Figure 5(c).
3. Algebraically regenerate each block using either linear (Figure 5(d)) or cubic-Hermite spline interpolation.[28] (Figure 5(e))
4. Using the approximate mesh, apply algebraic warping scheme to attenuate the remainder of the surface perturbations. See Figure 5(f).

Each finite-element in the "super mesh" uses linear shape functions, with its Young's modulus inversely proportional to its volume, V. We also scale the Young's modulus by the ratio of the longest cell edge to the shortest cell according to

$$E = \frac{1}{V}\frac{\max(\|L_i\|)}{\min(\|L_i\|)}, \tag{13}$$

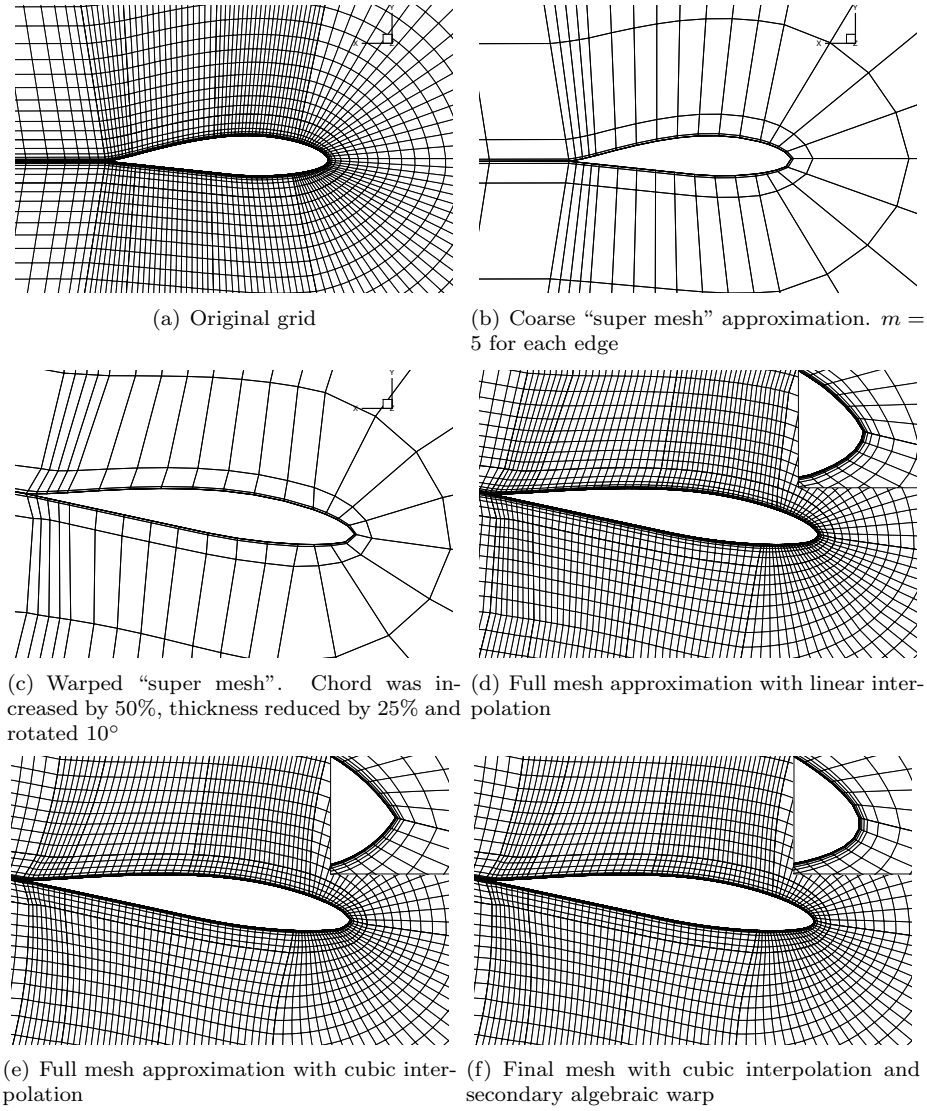
where L_i represents the length of the 12 edges in each element. Thus, for two elements with similar volumes, the highly elongated cell has a higher modulus. After applying the finite element method we obtain the following system of equations

$$\begin{bmatrix} K_{uu} & K_{us} \\ K_{su} & K_{ss} \end{bmatrix} \begin{bmatrix} u_u \\ u_s \end{bmatrix} = \begin{bmatrix} f_u \\ f_s \end{bmatrix}, \tag{14}$$

where the stiffness matrix K is partitioned according to the unknown (u subscript) and specified (s subscript) degrees of freedom. Since we enforce Dirichlet boundary conditions on the solid wall boundaries, the far field boundaries and the out-of-plane degree of freedom on a symmetry plane (if included), a significant number of the degrees of freedom are known. To find the unknown degrees of freedom we must solve

$$K_{uu}u_u = -K_{us}u_s. \tag{15}$$

We use PETSc[29–31] for the parallel assembly of the stiffness matrix. To solve Equation (15) we use parallel direct methods available through the PETSc interface: either Spooles[32] or SuperLU_DIST.[33] To minimize the amount of inter-processor communication during a mesh perturbation and during sensitivity analysis, we perform N_{wall} (Number of "super node" degrees of freedom on wall boundaries) back-solves on the factorized linear system of equations (Equation (15)) to determine $dX_{V_{FE}}/dX_{S_{FE}}$ — the derivative of the volume "super nodes" with respect to the "super nodes" on the wall boundaries. As presented in the section D, this incurs only a fixed, modest memory penalty. However, with $dX_{V_{FE}}/dX_{S_{FE}}$ computed, a



(a) Original grid

(b) Coarse "super mesh" approximation. $m = 5$ for each edge

(c) Warped "super mesh". Chord was increased by 50%, thickness reduced by 25% and rotated 10°

(d) Full mesh approximation with linear interpolation

(e) Full mesh approximation with cubic interpolation

(f) Final mesh with cubic interpolation and secondary algebraic warp

Figure 5: Hybrid mesh perturbation scheme

mesh perturbation and mesh sensitivity analysis can be computed in parallel without requiring the collective communication resulting from the solution of Equation (15).

C. Example

To demonstrate the robustness of the hybrid mesh warping scheme, we modify a structured multi-block grid from the 4^{th} AIAA Drag Prediction Workshop.[18] The specific grid is the wing-body-tail configuration known as DPW4, with the tail incidence angle of 0°. y^+ is approximately 1 corresponding to an off-wall spacing of 0.001478 in and a reference chord of 275.8 in. The structured mesh uses O-grids around the wing, fuselage and horizontal stabilizer. These small blocks render the single block algebraic scheme ineffective since even small perturbations exceed the full block dimension. A view of the surface mesh and symmetry plane are shown in Figure 6. The quality metric used for grid comparison purposes is the relative determinate:

$$Q = \frac{\min(\det J_i)}{\max(\det J_i)}, \tag{16}$$

where the i^{th} metric Jacobian, J_i, is evaluated at the corners and midpoints of each element in a $3 \times 3 \times 3$ stencil. We perform the following geometric modifications simultaneously using the linear FFD volumes in

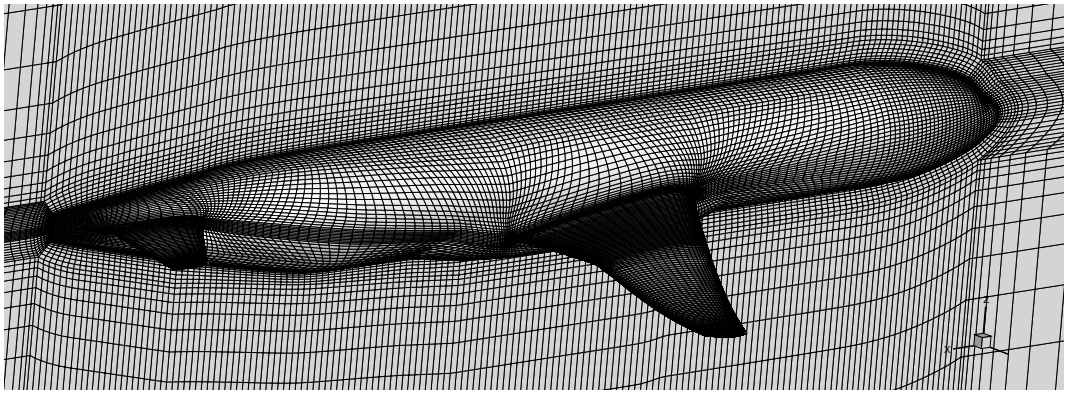


Figure 6: DPW4 surface grid

Figure 1.

- 296" increase in span; 25% semi-span
- A reduction in wing sweep to 0°
- A 100" vertical deflection at the tip; 8% semi-span
- -5°of additional twist at the tip

A planform view of the original and warped grids are shown in Figure 7. Figure 8 shows the quality metric distributions for the original and warped grids. The mesh perturbation has decreased the minimum quality from 0.249 to 0.233, however, the overall distribution remains practically unchanged.

D. Performance

The performance of the warping algorithm is of great importance for multi-disciplinary optimization. When performing an aerodynamic shape optimization, the grid must be perturbed for each change in design variable. For the case of aerostructural optimization, additional mesh perturbations must be performed for each load and displacement transfer. Depending on the solution scheme and the aerostructural solution tolerance, this can range from $\mathcal{O}(10)$ to $\mathcal{O}(100)$. Clearly, an efficient algorithm is a necessity. The hybrid mesh warping scheme achieves this by efficient parallelism with very little communication overhead. The only communication required for a mesh perturbation is the synchronization of surface perturbations across processors. The remainder of the solid warping and grid regeneration proceeds with no communication.

The parallel efficiency of the warping algorithm is shown in Figure 9. We take the 3.8M cell DPW4 grid with 4 "super nodes" on each edge. This corresponds to approximately 22,000 unknown degrees of freedom in the "super mesh" and perform the mesh perturbation on 1 to 128 processors. These timings were performed on the University of Toronto's SciNet GPC cluster, whose processors are connected with InfiniBand. The setup time includes one time procedures, such as loading the mesh into memory, partitioning the blocks, computing and factorizing the stiffness stiffness matrix, and performing N_S back-solves. The memory usage refers to the resident storage of $dX_{V_{FE}}/dX_{S_{FE}}$. The computational times indicate the parallel efficiency drops considerably with increasing number of processors. However, the mesh perturbation time still decreases with increasing number of processors and with a large number of processors, it is small enough to not significantly affect the overall optimization time. The setup, conversely scales quite poorly. This is an indication of the poor scaling of direct solution methods. However, since this must only be computed once for a given optimization problem, it is acceptable. Additionally, since $dX_{V_{FE}}/dX_{S_{FE}}$ is always the same for a given initial grid, it can be computed once and stored on disk. It must be noted the size of the "super mesh" does *not* necessarily scale with the with the size of the grid; it is directly related to the geometric complexity of the geometry and blocking topology. It is possible to use the same "super mesh" on a grid with the same blocking topology but with many more nodes. It is evident the relative memory overhead diminishes even further with larger grids and the same "super mesh".

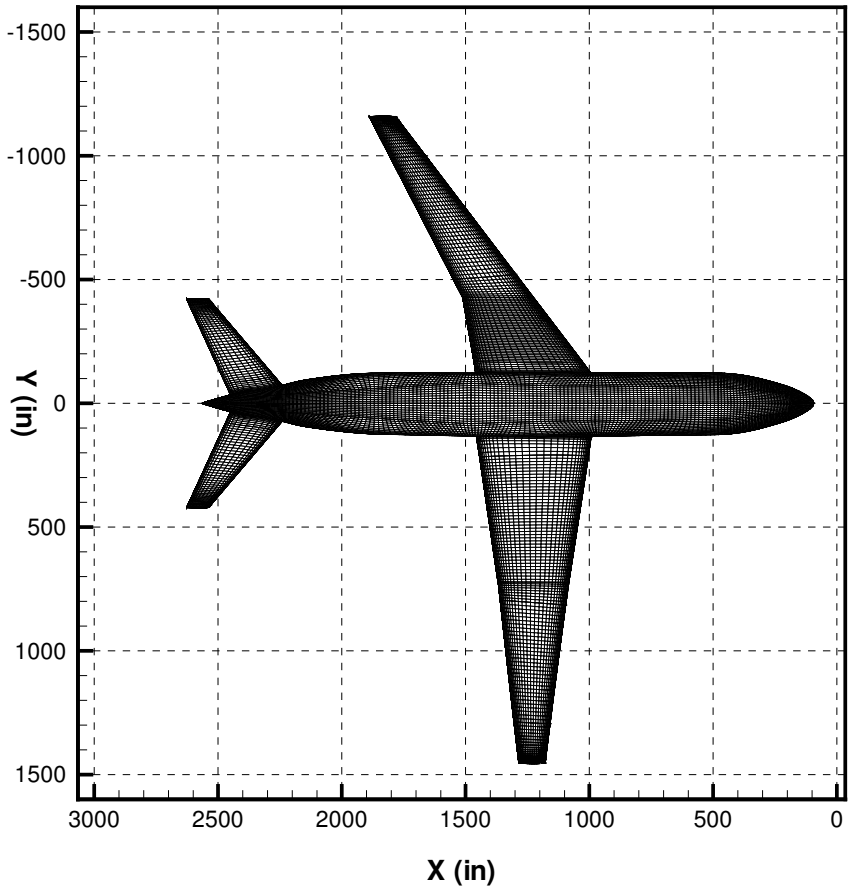


Figure 7: Original DPW4 planform (top); modified planform with un-swept wing (bottom)

1. *Grid Perturbation Sensitivity*

The evaluation of the sensitivity of the volume grid coordinates with respect to surface perturbations can be very costly. For MDO systems, this can be even more costly since the mesh sensitivity to the states of all disciplines must be considered in addition to the sensitivity with respect to the design variables. The adjoint and total sensitivity equations for a coupled aerostructural system are[34]

$$\begin{bmatrix} \frac{\partial \mathcal{A}}{\partial w} & \frac{\partial \mathcal{A}}{\partial u} \\ \frac{\partial \mathcal{S}}{\partial w} & \frac{\partial \mathcal{S}}{\partial u} \end{bmatrix}^T \begin{bmatrix} \psi_{\mathcal{A}} \\ \psi_{\mathcal{S}} \end{bmatrix} = - \begin{bmatrix} \frac{\partial I}{\partial w} \\ \frac{\partial I}{\partial u} \end{bmatrix}, \tag{17}$$

where $\mathcal{A}$ and w are the aerodynamic residuals and states, $\mathcal{S}$ and u are the structural residuals and states and I is a function of interest. Once an adjoint vector for a given function of interest is computed, the total sensitivity can be found using

$$\frac{\mathrm{d}I}{\mathrm{d}x_{dv}} = \frac{\partial I}{\partial x_{dv}} + \psi_{\mathcal{A}}^T \frac{\partial \mathcal{A}}{\partial x_{dv}} + \psi_{\mathcal{S}}^T \frac{\partial \mathcal{S}}{\partial x_{dv}}. \tag{18}$$

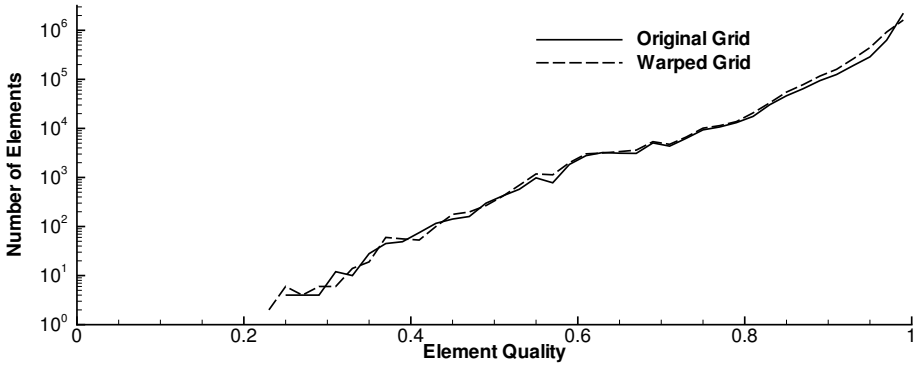


Figure 8: Quality comparison between original and warped grids

Processors	Setup Time (s)	Warp Time (s)	Parallel Efficiency (Warp)	Memory (MB/Proc)
1	260	2.00	100%	1127
2	177	1.45	69.0%	563.5
4	134	0.907	55.1%	281.8
8	117	0.53	47.2%	140.9
16	132	0.448	27.9%	70.4
32	208	0.275	22.7%	35.2
64	495	0.169	18.5%	17.6
128	702	0.146	10.7%	8.8

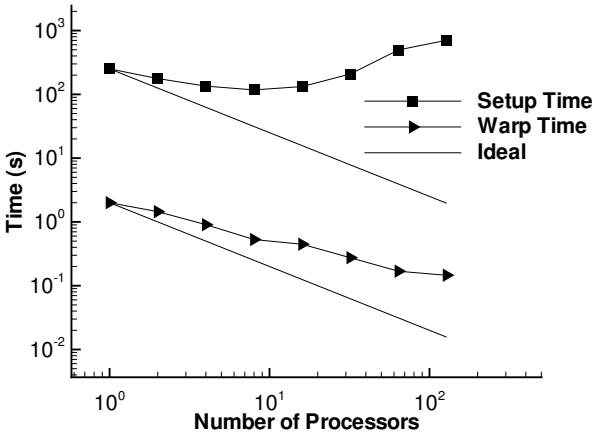


Figure 9: Strong scaling mesh perturbation results

When expanded, the sensitivity term, $\partial X_V/\partial X_S$, is computed using the following matrix-vector products:

$$\left[\frac{\partial \mathcal{A}}{\partial u}\right]^T \psi_{\mathcal{A}} = \left[\frac{\partial X_S}{\partial u}\right]^T \left[\frac{\partial X_V}{\partial X_S}\right]^T \left[\frac{\partial \mathcal{A}}{\partial X_V}\right]^T \psi_{\mathcal{A}} \tag{19}$$

$$\left[\frac{\partial \mathcal{A}}{\partial x_{\text{dv}}}\right]^T \psi_{\mathcal{A}} = \left[\frac{\partial X_S}{\partial x_{\text{dv}}}\right]^T \left[\frac{\partial X_V}{\partial X_S}\right]^T \left[\frac{\partial \mathcal{A}}{\partial X_V}\right]^T \psi_{\mathcal{A}} \tag{20}$$

$$\left[\frac{\partial I}{\partial x_{\text{dv}}}\right]^T = \left[\frac{\partial X_S}{\partial x_{dv}}\right]^T \left[\frac{\partial X_V}{\partial X_S}\right]^T \left[\frac{\partial I}{\partial X_V}\right]. \tag{21}$$

$\partial X_V/\partial X_S$ is a very large matrix ($\mathcal{O}(10^6)$ rows, $\mathcal{O}(10^4)$ columns for a 10^6 node mesh); however, it is extremely sparse. To compute the matrix-transpose product $\partial X_V/\partial X_S$ with a vector V (either $[\partial \mathcal{A}/\partial X_V]^T\psi_{\mathcal{A}}$ or $\partial I/\partial X_V$) we take a two-step procedure. First we recognize that the "super nodes" on the mesh surface have a global mesh dependence; the corresponding column in $\partial X_V/\partial X_S$ is dense. Second, the rest of the nodes or "regular nodes" only have a local block dependence, and generally only a single mesh line radiating from a perturbed surface.

The strategy we use here is to compute and store $\partial X_V/\partial X_S$ using Tapenade Automatic Differentiation in forward mode for the "regular nodes" only.[35] For the "super nodes" the stored, pre-computed $dX_{V_{FE}}/dX_{S_{FE}}$ is the derivative of the "super nodes" with respect to the "super nodes" on the surface is used with an automatically differentiated version of the of the tri-cubic interpolation scheme taking, the dot-product of V is taken with each "super node" column of $dX_{V_{FE}}/dX_{S_{FE}}$ as it is computed.

As with the warping scheme itself, the stored $dX_{V_{FE}}/dX_{S_{FE}}$ derivative allows the computations to occur in parallel with the only inter-processor communication occurring during the final assembly of the resulting matrix transpose product.

IV. Discipline Solvers

For the CFD solver, we use SUmb,[36] a finite-volume, cell-centred multiblock solver for the Reynolds-averaged Navier–Stokes equations. This solver was developed at Stanford University under the sponsorship of the Department of Energy. A more complete description of the code, including the adjoint implementation can be found in Mader et. al.[37]

The structural solver utilized in this work is the Toolbox for the Analysis of Composite Structures (TACS).[22] It is a parallel finite-element solver specifically written for the analysis and optimization of composite structures. The code is written in C++ and contains a Python interface to facilitate integration in multi-disciplinary problems. The input to TACS is handled through an interface written in Python. This interface allows for easy modification of algorithms and flexibility in building finite-element models. It also provides access to a powerful scripting language that can perform operations that an input-file approach could not achieve. Geometry is handled through an abstract layer that can be extended to allow input from many different sources.

The design variables in TACS are handled in a distributed manner and their dependence is automatically determined by TACS after the finite-element model has been assembled. Design variables may include both geometry and material-type variables such as thickness and ply angles, etc. Sensitivities of analysis outputs may be determined using either an adjoint or a direct method. Additionally, sensitivities with respect to eigenvalues may be computed. All derivatives involved in the sensitivity calculations are computed analytically. As a result, TACS can compute sensitivities accurately and efficiently.

V. Coupling Methodology

The load and displacement transfer implementation follows the work of Brown,[38] later employed by Martins et al.[39] In this transfer scheme, rigid links are used to extrapolate the displacements from the structural surface to the outer mold line (OML) of the CFD wetted surface. These rigid links are constructed between the aerodynamic surface mesh points and the points on the structural model lying closest to this set of points. The consistent force vector is determined by employing the method of virtual work, ensuring that the force transfer is conservative. The integration of the forces occurs on the aerodynamic mesh and are transmitted back through the rigid links to the structure.

This scheme has two primary advantages: it is consistent and conservative by construction and it may be used to transfer loads and displacements between aerodynamic and structural meshes that are not coincident. Thus, structural models that do not conform exactly with the OML may still be easily analyzed. The disadvantage of this method is that it may result in large point moments. We minimize this effect by using a structure surface that is very nearly coincident with the CFD wetted surface.

A. Optimization

Due to the complex and costly nature of aerostructural analysis, it is desirable to use a gradient-based optimizer to reduce the number of function evaluations necessary to reach a local optimum. Ideally, we aim to use a coupled-adjoint formulation,[39] however, since the implementation is currently incomplete we use finite-differencing to determine the function gradients. For this work we use SNOPT, an optimizer based on the SQP approach.[40]

VI. Results

In this section we present preliminary high-fidelity MDO results using the FFD parametrization approach and the hybrid mesh warping scheme. The test case used for the optimization is mostly academic in nature, but nevertheless serves to highlight important design trade-offs that can be captured using high-fidelity MDO.

The choice of a multidisciplinary objective is not immediately obvious. For a strictly aerodynamic

optimization, we may wish to fix the lift coefficient and minimize the drag or maximize the lift to drag ratio. For the structural optimization problem, we may want to minimize the structural mass, subject to a maximum stress constraint. A lift-constrained drag minimization objective may be used for an aerostructural optimization problem, however, such an objective fails to account for the fuel savings resulting from a lower empty weight. One multidisciplinary objective that is a comprise between these choices is the cruise-climb mode of the Breguet Range equation

$$R = \frac{V}{c}\frac{L}{D}\ln\left(\frac{W_1}{W_2}\right), \tag{22}$$

where R is the range, V is the flight speed, c is the thrust-specific fuel consumption, L is the lift, D is the drag and W_1 and W_2 are the initial and final cruise weights, respectively. The effect of the aerodynamic performance is captured in the L/D term and the structural performance in the uniform reduction of W_1 and W_2 due to a lower structural mass. There are still however, complex multidisciplinary interactions when the constraints are considered. For the optimization results presented herein we use a fixed Mach number and the thrust-specific fuel consumption is assumed fixed. We therefore only maximize $L/D\ln(W_1/W_2)$. The starting point for the optimization is a high aspect-ratio, slightly swept wing typified by a turbo-prop transport aircraft. The geometry is loosely based on a Bombardier Q400 wing.[41] A summary of the basic geometric properties are given in Table 1. We assume that the initial wing mass is 20% of the Operational Empty Weight (OEW) and the remainder of the OEW remains fixed. A summary of the weights used for the optimization are given in Table 2.

Table 1: Geometric properties of the baseline wing

Parameter	Value
Area	63.1 m^2
Span	28.4 m
Aspect Ratio	12.8
Sweep	10°
Taper Ratio	0.41
Twist	0°
Dihedral	2.5°
Root Airfoil	RAE 2282
Tip Airfoil	RAE 2282

Table 2: Weights for Breguet range equation

Parameter	Value
Maximum Takeoff Weight	29,260 kg
Operational Empty Weight	17,185 kg
Fixed Structural Weight	13,748 kg
Payload	6,9853 kg
Fuel	5,090 kg
W_1	25,823 + m_{wing} kg
W_2	20,733 + m_{wing} kg

Since we use finite-differences to compute gradients, we use a relatively small set of design variables. Two flight conditions are considered: a cruise condition and a maneuver condition. The range is computed for the cruise condition while the von Mises stress constraint is computed at the 2.5g maneuver condition. An angle of attack is specified at each condition. The cruise condition is Mach 0.78 at 41,000 ft, while the maneuver condition is Mach 0.95 at 35,000 ft. The maneuver is performed at the design dive speed. The geometric design variables are the span, sweep, and twist. The span design variable is tied to the wing chord in such a way that the wing area remains constant thus effectively changes the wing's aspect ratio. Two structural design variables are used: the skin and spar thickness at the root. A linearly varying thickness distribution is assumed terminating in a 1.5 mm gauge thickness at the wing tip. A description of each design variable, their starting value and bounds is given in Table 4. Three constraints are enforced: A lift constraint for the cruise condition to ensure lift and weight are equal; a second lift constraint for the maneuver condition, with the lift equal to 2.5 times the weight; and a constraint on the maximum von Mises stress of the structure. The value is set at 2.0 which can be interpreted as the minimum safety before failure. The Kreisselmeier–Steinhauser (KS) function is used to combine all stresses into a single constraint.[42–44] The optimization constraints are listed in Table 3.

A representative rib-spar-skin structure is generated using `pyLayout` consisting of two spars and 24 ribs. We assume an isotropic material property with a yield stress of 476MPa. The CFD mesh contains approximately 440,000 hexahedral cells, while the structural mesh contains 11,405 nodes, corresponding to 68,430 degrees of freedom. The initial grid along with the Mach contours are shown in Figure 10. Figrure 11

Table 3: Optimization constraints

Parameter	Lower Bound	Upper Bound
$L_{\text{cruise}} - W$	0	0
$L_{\text{maneuver}} - 2.5 \times W$	0	0
KS_{vms}	2.0	–

shows the internal structure of the wing along with the von Mises stress contours at the optimization initial condition. A section of the upper skin of the wing structure is removed to show the internal structure. Note that the finite-element mesh shows only the edges of the fourth-order shell elements and not the interior nodes.

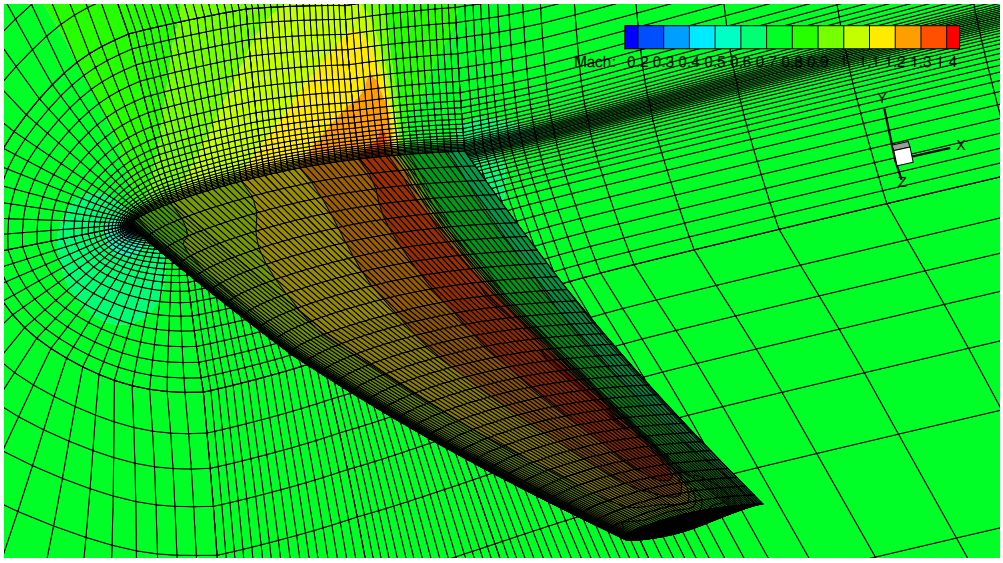


Figure 10: Grid and Mach contours for initial design

A comparison of the initial and optimized design variables is given in Table 4 and a convergence history plot of the range objective, KS constraint and the wing structural weight is shown in Figure 12.

Table 4: Optimization results

Parameter	Initial Value	Optimized Value	Lower Bound	Upper Bound
α_{cruise}	2.5°	5.2°	-12.5°	12.5°
α_{maneuver}	6.25°	9.8°	-12.5°	12.5°
Span	1	0.83	0.5	1.25
Sweep	10^o	25.5°	0^o	45°
Twist (Mid span)	0°	2.3°	-10°	10°
Twist (Tip)	0°	5.5°	-10°	10°
Root Skin	13.7 mm	14.0 mm	1.5 mm	1000 mm
Root Spar	6.9 mm	6.6 mm	1.5 mm	1000 mm

The optimized sweep reflects an important trade-off in transonic wing design: highly swept wings have lower drag, but require a heavier structure. We would expect from a strictly aerodynamic perspective that the wing should sweep to its aft limit of 45°, but the moderating effect of the weight penalty in the Breguet range equation has limited the sweep to a more conventional 25°. The wing aspect ratio has decreased

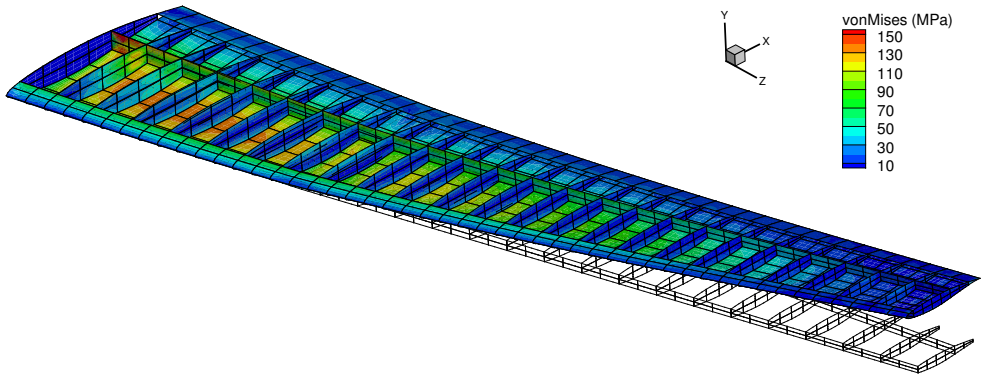


Figure 11: Structure and von Misis stress for initial design. Outline is undisplaced jig shape.

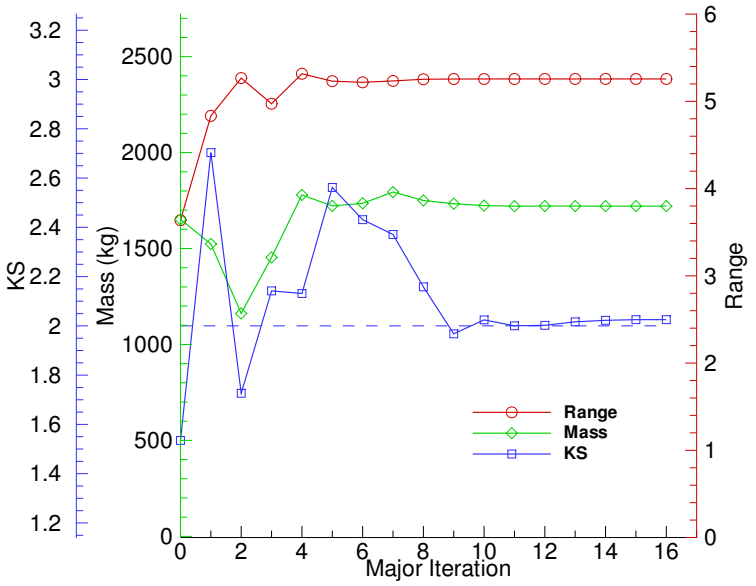


Figure 12: Optimization convergence history

from 12.8 to 8.81, again reducing the aerodynamic efficiency (higher induced drag) but resulting in a lighter structure. There is also significant amount of twist (washout) towards to the tip which serves to reduce the tip loading and results in span-wise lift distribution that reduces bending moments, thus lowering the structural mass. Figure 13 shows a comparison of the Mach number contours on the starting planform and the optimized planform along the with chord-wise C_P distributions. We can see the optimized wing has reduced the normal shock strength on the upper surface of the wing, but has not been able to eliminate it completely. We would expect with the addition of local shape modification design variables, a further significant reduction in drag can be archived by further reducing the shock strength. Figure 14 shows an exploded view of the structure and the Mach number contours at the maneuver condition. We can see the stress is highest at the root of the wing skins and the root of the rear spar.

VII. Conclusion

In this paper we presented a CAD-free geometry parametrization technique based on the free-form deformation volume approach. We show this approach is well suited to high-fidelity multidisciplinary optimization

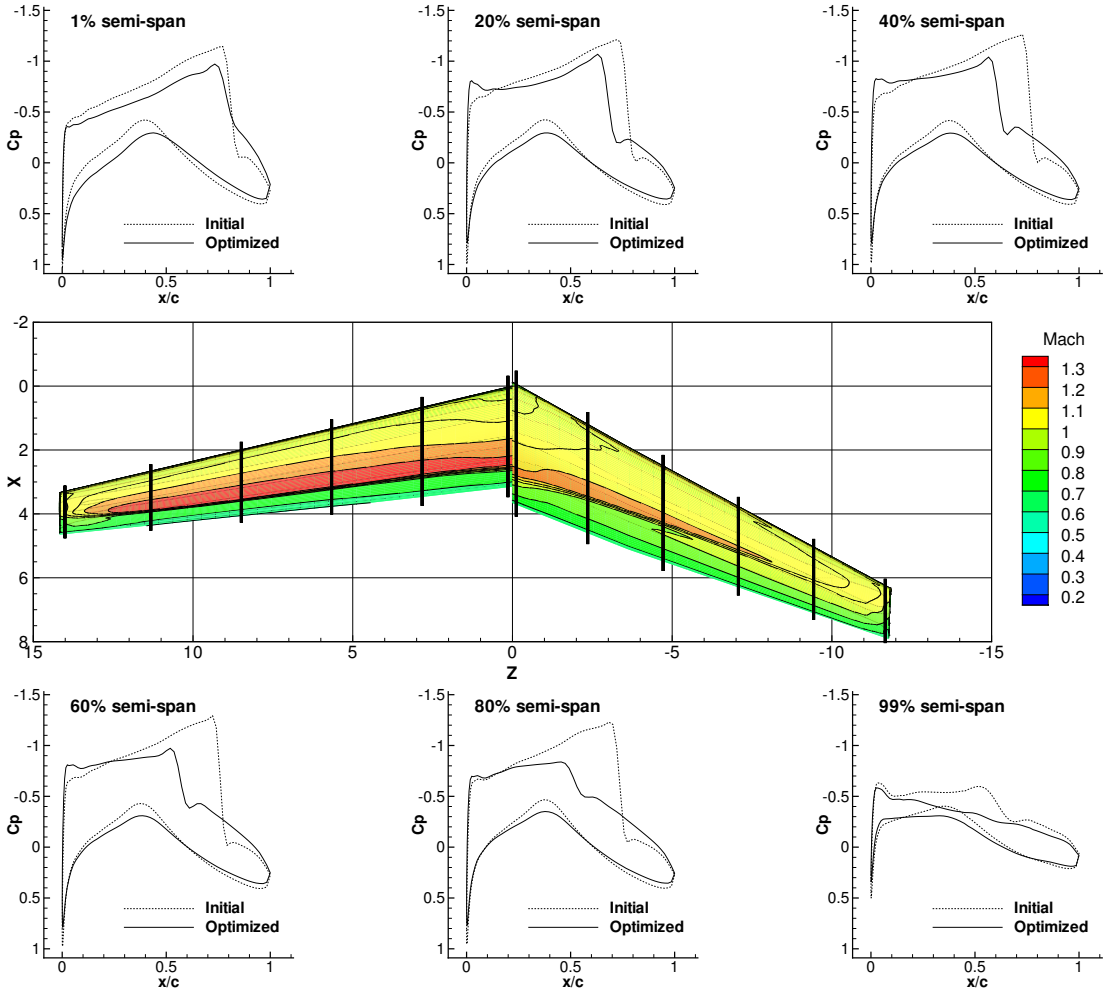


Figure 13: Planform comparison between initial (left) and optimized (right) geometries and pressure coefficient distributions at 6 parametric spanwise locations

problems. We introduced a parallel, hybrid mesh perturbation scheme combining a linear-elasticity method with an fast algebraic perturbation algorithm. High quality perturbed grids can be computed. A model high-fidelity aerostructural problem is presented showing the redesign of a sub-sonic wing for transonic flow conditions. We are able to capture the multidisciplinary trade-off between sweep and structural weight which is important in this flight regime.

Acknowledgments

The authors gratefully acknowledge financial assistance from the Natural Sciences and Engineering Research Council (NSERC). Computations were performed on the General-Purpose Cluster supercomputer at the SciNet HPC Consortium. SciNet is funded by: the Canada Foundation for Innovation under the auspices of Compute Canada; the Government of Ontario; Ontario Research Fund—Research Excellence; and the University of Toronto.

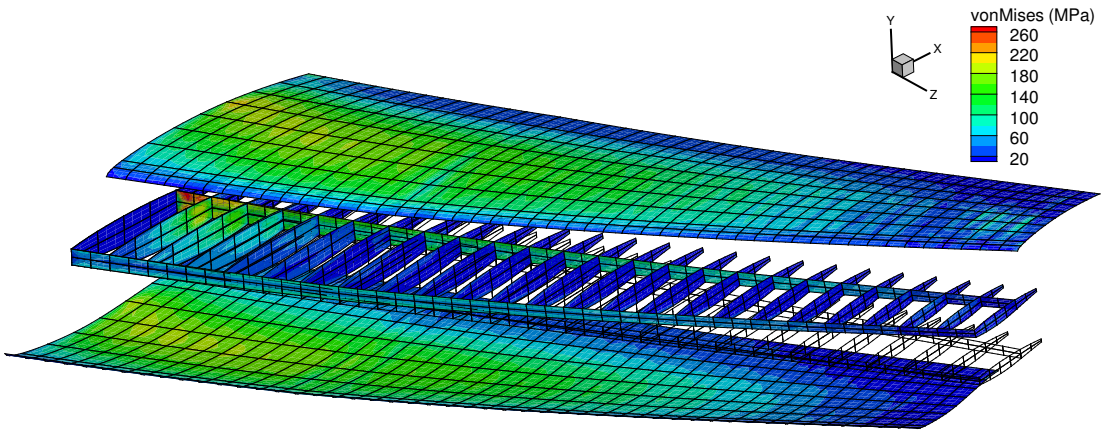


Figure 14: Maneuver load condition

References

[1]Moore, G. E., "Cramming more components onto integrated circuits," *Electronics*, Vol. 38, No. 8, April 1965.

[2]Green, J., "Greener by Design — The technology challenge," *The Aeronautical Journal*, Feb. 2002.

[3]Birch, N., "2020 Vision: The Prospects for Large Civil Aircraft Propulsion," *The Aeronautical Journal*, Vol. 104, No. 1038, August 2000, pp. 347–352.

[4]Reuther, J. J., Jameson, A., Alonso, J. J., , Rimlinger, M. J., and Saunders, D., "Constrained Multipoint Aerodynamic Shape Optimization Using an Adjoint Formulation and Parallel Computers, Part 1," *Journal of Aircraft*, Vol. 36, No. 1, 1999, pp. 51–60.

[5]Reuther, J. J., Jameson, A., Alonso, J. J., , Rimlinger, M. J., and Saunders, D., "Constrained Multipoint Aerodynamic Shape Optimization Using an Adjoint Formulation and Parallel Computers, Part 2," *Journal of Aircraft*, Vol. 36, No. 1, 1999, pp. 61–74.

[6]Zingg, D. W., Leung, T. M., Diosady, L., Truong, A. H., Elias, S., and Nemec, M., "Improvements to a Newton-Krylov Adjoint Algorithm for Aerodynamic Optimization," *17th AIAA Computational Fluid Dynamics Conference*, 2005.

[7]Nadarajah, S. K. and Jameson, A., "Optimum Shape Design for Unsteady Three-Dimensional Viscous Flows Using a Nonlinear Frequency-Domain Method," *Journal of Aircraft*, Vol. 44, No. 5, 2007, pp. 1513–1527.

[8]Jameson, A., "Aerodynamic design via control theory," *Journal of Scientific Computing*, Vol. 3, No. 3, 1988, pp. 233–260.

[9]Chittick, I. R. and Martins, J. R. R. A., "Aero-Structural Optimization Using Adjoint Coupled Post-Optimality Sensitivities," *Structures and Multidisciplinary Optimization*, 2007.

[10]Liebeck, R. H., "Design of the Blended Wing Body Subsonic Transport," *Journal of Aircraft*, Vol. 41, No. 1, 2004, pp. 10–25.

[11]Wakayama, S. and Kroo, I., "The Challenge and Promise of Blended Wing Body Optimization," *AIAA Paper 98−4736*, 1998.

[12]Kafyeke, F., Abdo, M., Pépin, F., Piperni, P., and Laurendeau, E., "Challenges of Aircraft Design Integration," *RTO AVT Symposium on "Reduction of Military Vehicle Acquisition Time and Cost through Advanced Modelling and Virtual Simulation"*, 2002.

[13]Cramer, E. J., Dennis, J. E., Frank, P. D., Lewis, R. M., and Shubin, G. R., "Problem formulation for multidisciplinary optimization," *SIAM Journal on Optimization*, Vol. 4, 1994, pp. 754–776.

[14]Sobieszczanski-Sobieski, J. and Haftka, R., "Multidisciplinary aerospace design optimization: Survey of recent developments," *Structural Optimization*, Vol. 14, No. 1, 1997, pp. 1–23.

[15]Irons, B. M. and Tuck, R. C., "A Version of the Aitken Accelerator for Computer Iteration," *International Journal for Numerical Methods in Engineering*, Vol. 1, 1969, pp. 275–277.

[16]Samareh, J. A., "Survey of Shape Parameterization Techniques for High-Fidelity Multidisciplinary Shape Optimization," *AIAA Journal*, Vol. 39, 2001, pp. 877–884.

[17]Peterson, P., Martins, J. R. R. A., and Alonso, J. J., "Fortran to Python Interface Generator with an Application to Aerospace Engineering," *Proceedings of the 9th International Python Conference, Long Beach, CA*, 2001.

[18]"4th AIAA CFD Drag Prediction Workshop," Internet, June 2009, http://aaac.larc.nasa.gov/tsab/cfdlarc/aiaa-dpw/.

[19]Sederberg, T. W. and Parry, S. R., "Free-Form Deformation of Solid Geometric Models," *SIGGRAPH Comput. Graph.*, Vol. 20, 1986, pp. 151–160.

[20]Turk, G. and Levoy, M., "Zippered Polygon Meshes from Range Images," *Computer Graphics Proceedings*, 1994, pp. 311–318, http://graphics.stanford.edu/data/3Dscanrep/.

[21]Martins, J. R. R. A., Sturdza, P., and Alonso, J. J., "The Complex-Step Derivative Approximation," *ACM Transactions on Mathematical Software*, Vol. 29, No. 3, 2003, pp. 245–262.

[22]Kennedy, G. J. and Martins, J. R. R. A., "Parallel Solution Methods for Aerostructural Analysis and Design Optimization," *Proceedings of the 13th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, Fort Worth, Texas, September 2010.

[23]Batina, J. T., "Unsteady Euler Airfoil Solutions Using Unstructured Dynamic Meshes," *AIAA Journal*, Vol. 28, 1990, pp. 1381–1388.

[24]Degand, C. and Farhat, C., "A three-dimensional torsional spring analogy method for unstructured dynamic meshes," *Computers and Structures*, Vol. 80, 2002, pp. 305–316.

[25]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, 2008, pp. 1695–1704.

[26]Hicken, J. and Zingg, D., "Aerodynamic Optimization Algorithm with Integrated Geometry Parameterization and Mesh Movement," *AIAA Journal*, Vol. 48, No. 2, February 2009, pp. 400–413.

[27]Reuther, J., Jameson, A., Famer, J., Martinelli, L., and Saunders, D., "Aerodynamic Shape Optimization of Complex Aircraft Configurations via an Adjoint Formulation," *AIAA 34th Aerospace Sciences Meeting and Exhibit*, 1996.

[28]Lekien, F. and Marsden, J., "Tricubic interpolation in three dimensions," *International Journal for Numerical Methods in Engineering*, Vol. 63, 2005, pp. 455–471.

[29]Balay, S., Buschelman, K., Gropp, W. D., Kaushik, D., Knepley, M. G., McInnes, L. C., Smith, B. F., and Zhang, H., "PETSc Web page," 2009, http://www.mcs.anl.gov/petsc.

[30]Balay, S., Buschelman, K., Eijkhout, V., Gropp, W. D., Kaushik, D., Knepley, M. G., McInnes, L. C., Smith, B. F., and Zhang, H., "PETSc Users Manual," Tech. Rep. ANL-95/11 - Revision 3.0.0, Argonne National Laboratory, 2008.

[31]Balay, S., Gropp, W. D., McInnes, L. C., and Smith, B. F., "Efficient Management of Parallelism in Object Oriented Numerical Software Libraries," *Modern Software Tools in Scientific Computing*, edited by E. Arge, A. M. Bruaset, and H. P. Langtangen, Birkhäuser Press, 1997, pp. 163–202.

[32]Ashcraft, C. and Grimes., R., "SPOOLES: An object-oriented sparse matrix library," *Ninth SIAM Conference on Parallel Processing*, 1999.

[33]Li, X. S. and Demmel, J. W., "SuperLU DIST: A scalable distributed-memory sparse direct solver for unsymmetric linear system," *ACM Trans. Mathematical Software*, Vol. 29, No. 2, 2003, pp. 110–140.

[34]Martins, J. R. R. A., Martins, J. R. R. A., Alonso, J. J., Alonso, J. J., Reuther, J. J., and Reuther, J. J., "High-Fidelity Aero-Structural Design Optimization of a Supersonic Business Jet," *Journal of Aircraft*, Vol. 41, No. 3, 2004, pp. 523–530.

[35]Hascoët, L. and Pascual, V., "TAPENADE 2.1 User's Guide," Technical report 300, INRIA, 2004.

[36]van der Weide, E., Kalitzin, G., Schluter, J., and Alonso, J. J., "Unsteady Turbomachinery Computations Using Massively Parallel Platforms," 2006, AIAA 2006-0421.

[37]Mader, C. A., Kenway, G. K. W., and Martins, J. R. R. A., "Toward High-Fidelity Aerostructural Optimization Using a Coupled ADjoint Approach," *Proceedings of the 12th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, Victoria, BC, 2008, AIAA 2008-5968.

[38]Brown, S., "Displacement extrapolation for CFD+CSM aeroelastic analysis," AIAA Paper 97-1090, 1997.

[39]Martins, J. R. R. A., Alonso, J. J., and Reuther, J. J., "A Coupled-Adjoint Sensitivity Analysis Method for High-Fidelity Aero-Structural Design," *Optimization and Engineering*, Vol. 6, No. 1, March 2005, pp. 33–62.

[40]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.

[41]Bombardier, "Q400 - Specifications," http://www2.bombardier.com/q400/en/specifications.jsp, August 2010.

[42]Raspanti, C., Bandoni, J., and Biegler, L., "New strategies for flexibility analysis and design under uncertainty," *Computers and Chemical Engineering*, Vol. 24, 2000, pp. 2193–2209.

[43]Wrenn, G., "An indirect method for numerical optimization using the Kreisselmeier-Steinhauser function," NASA Technical Report CR-4220, 1989.

[44]Poon, N. M. K. and Martins, J. R. R. A., "An Adaptive Approach to Constraint Aggregation Using Adjoint Sensitivity Analysis," *Structures and Multidisciplinary Optimization*, Vol. 30, No. 1, 2007, pp. 61–73.