Geometry and Structural Modeling for High-Fidelity Aircraft Conceptual Design Optimization

John T. Hwang, [*] Gaetan K. W. Kenway, [†] and Joaquim R. R. A. Martins[‡]

University of Michigan, Ann Arbor, Michigan, 48109, United States

High-fidelity computational design tools have advanced considerably in the last few decades, but there are still bottlenecks that suppress their impact in conceptual design. This paper aims to address some of these bottlenecks through a parametric geometry modeler for unconventional configurations with an efficient derivative computation and a parametric structural modeler that automatically generates full finite element meshes. As a demonstration, lift-constrained drag minimization of a truss-braced wing design is performed with respect to 532 design variables and 951 constraints with the Euler equations solved on a 1.42 million cell mesh.

I.   Introduction

OVER the last five decades, the fuel efficiency of commercial aircraft has approximately doubled through advancements in technology and design [1]. However, this trend has stagnated in recent years, as each aircraft becomes more optimized and further improvements become more technically challenging to achieve. This concern is compounded by the fact that air traffic growth is expected to outpace efficiency improvements in the next two decades [2], in the face of rising fuel costs and growing environmental concerns.

At a fundamental level, aircraft design improvements target either the propulsion system by lowering thrust specific fuel consumption or the airframe by reducing structural weight and aerodynamic drag. On the airframe side, computational design is an important tool that combines the mechanical efficiency of computers with the creativity and experience of the human designer. Numerical optimization has the potential to achieve the advances that were not within reach in the past, particularly by rapidly exploring revolutionary designs in which prior knowledge is limited.

With current practices, there is a disconnect because higher-fidelity, and thus more accurate, computational models are used later in the design process, when the potential for improvements is smaller. The aircraft design process has three stages, which are summarized here based on the treatment in Raymer [3]. Conceptual design involves the design of the configuration and layout through conceptual sketching, 'first-order' sizing, decisions on technologies, and simple optimization based on estimates of takeoff weight, L/D, etc. At preliminary design, the configuration is frozen and specialists perform compartmentalized analyses within each discipline. Detail design involves analysis and design of small-scale components such as ribs and spars as well as fabrication and testing. In computational design, many high-fidelity analysis and optimization algorithms are located in the preliminary to detail design stages of the process, but it would be desirable to be able to apply these algorithms earlier in the design process where they can have a much broader impact.

Naturally, the high-level research objective is then to enable high-fidelity multidisciplinary design optimization (MDO) in conceptual design, which has three implications. First, a parametric representation of the aircraft is needed to model the geometry and structure. Due to the difficulties of mesh generation, most high-fidelity tools can only handle small-scale design changes; however, conceptual design requires full versatility to compare configurations and radically different designs. Second, the analysis tools are difficult to integrate with each other when large, complex high-fidelity software is involved. In conceptual design, the breadth of designs considered make interdisciplinary coupling important, necessitating a multidisciplinary simulation for accuracy. Third, computational design is an iterative process that requires fluid user-computer interaction. High-fidelity tools often require a non-trivial amount of time to set up a simulation and execution is also very expensive, but conceptual design requires a fast turnaround time.

The structure of this paper is as follows. First, the overall approach will be described: analytic gradient-based optimization with a central geometry driving the aerodynamic and structural meshes. The second section will present a geometry modeler with emphasis on supporting multiple configurations, efficiently computing derivatives as a sparse Jacobian, and a simple parametrization that spans global to local shape changes. The third section will describe a detailed structural modeling tool built on a novel unstructured quadrilateral mesh generation algorithm. The fourth

---

[*]Ph.D. Candidate, Department of Aerospace Engineering, AIAA Student Member
[†]Postdoctoral Research Fellow, Department of Aerospace Engineering, AIAA Member
[‡]Associate Professor, Department of Aerospace Engineering, AIAA Associate Fellow

American Institute of Aeronautics and Astronautics

section will show optimization results of a truss-braced wing configuration based on the Boeing SUGAR Volt concept. Finally, the contributions and results will be summarized, and in particular, the ideas from the paper that have general applicability and utility will be highlighted.

II. Overview

Before presenting the contributions and results, this section will describe the high-level approach, the pieces of software, and the usage process.

A. Approach

MOTIVATION   The motivation for this project is the success and promise shown by modern high-fidelity MDO algorithms [4, 5]. For instance, Kenway and Martins' recently developed algorithm performed multi-point aerostructural optimization of the NASA Common Research Model (CRM) wing-body-tail configuration involving computational fluid dynamics (CFD) and finite element analysis (FEA) [4]. This algorithm coupled the Euler equations discretized on a 2 million cell mesh with a shell-element structure with 300,000 degrees of freedom, with nearly 500 aerodynamic shape and structural sizing design variables. Minimizing fuel burn resulted in a higher-span, lower-sweep design, while minimizing takeoff gross weight found a raked wingtip after optimization. The high-level approach adopted here is largely driven by the requirements of high-fidelity MDO algorithms such as that of Kenway and Martins.

GRADIENT-BASED OPTIMIZATION   Assuming efficient solvers are in place for each discipline, the primary challenge is handling the large number of design variables that are needed to take advantage of the accuracy and fine resolution of the CFD and FEA solvers. Due to the significant cost of each analysis, quasi-Newton gradient-based optimization is the only feasible method as the number of iterations scales linearly, or better in practice, with the number of design variables. The adjoint method enables computation of derivatives at a cost nearly independent of the number of design variables, so together, the adjoint method and quasi-Newton optimization can handle large-scale optimization problems with tremendous efficiency as demonstrated in the solution of 25,000 design-variable small satellite MDO problem [6].

Since a gradient-based approach is used, all components of the multidisciplinary computational model must be differentiable and be able to provide derivatives of its outputs with respect to its inputs. The derivatives must be efficiently computed and preferably with accuracy to numerical precision—inaccurate derivative information causes the optimizer to take more steps and limits the tolerance to which the optimization problem can be converged. Moreover, since the adjoint method solves a linear system with the transposes of Jacobian matrices, each component must be able to provide the transpose of its Jacobian of derivatives. For a more detailed discussion on the adjoint method, the reader is referred to Martins and Hwang [7].

DISCIPLINES   Figure 1 shows a breakdown of the disciplines in aircraft design. At the top level, one can separate the airframe and propulsion systems as well as mission analysis. Mission analysis can be further subdivided into aircraft allocation at the airline level, which chooses the mission, and the aircraft performance for a given mission. Emissions naturally falls under propulsion while noise is also in this category since the engine is usually the principal source of noise. On the airframe side, stability constraints and loads are both evaluated based on the aerodynamic analysis, while weights and materials are related to the structural analysis. The focus of this paper is on the meshing, analysis, and coupling of the aerodynamics and structures disciplines. The approach taken is designed with the other disciplines in mind, but they are not explicitly addressed in this paper.

GEOMETRY-CENTRIC IMPLEMENTATION   As mentioned in Sec. I, a parametric representation of the aircraft is one of the necessary components of computational design. Following from two of the authors' previous work [8], a geometry-centric approach is adopted where aircraft shape design variables parametrize the outer mold line (OML). The OML is represented using the B-spline engine presented in Hwang and Martins [8], which provides a continuous and optionally smooth surface description. B-splines have been used to model aircraft geometry in other previous work as well [9, 10]. The aerodynamic and structural surface meshes are mapped linearly from the B-spline control points, and the full aerodynamic and structural meshes are obtained using the CFD mesh movement algorithm and structural modeler, respectively. In summary, this approach allows shape variables to control a common geometry description which simultaneously drives the changes to the aerodynamic and structural meshes as the shape design variables change.

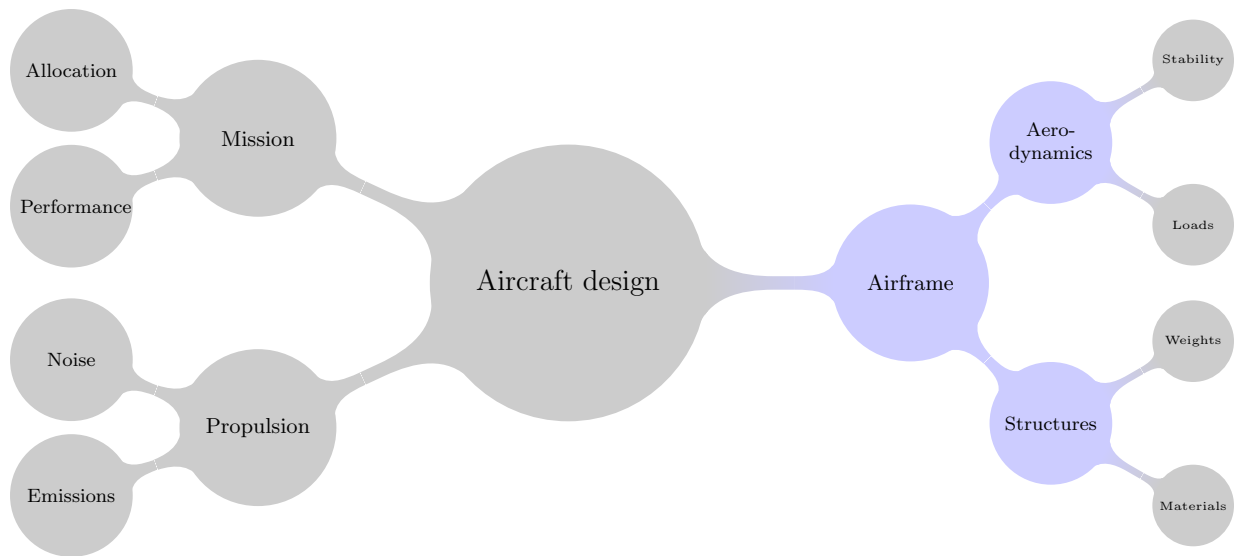American Institute of Aeronautics and Astronautics

Figure 1: Mind map of aircraft design disciplines. The circles in light blue are the focus of this paper.

B. Software

The algorithms developed in this paper will be integrated with several existing pieces of software for the demonstration of high-fidelity MDO in a conceptual design setting. They are shown in Fig. 2 in the context of aerostructural optimization. This figure uses the extended design structure matrix (XDSM) format [11] which represents components (or software in this case) along the diagonal and off-diagonal entries show dependence where data in the (i, j)th position is passed from the ith component to the jth component. The components related to geometry are in blue, those related to the CFD analysis are in red, and those related to FEA are in green. Some of the components are written in a compiled language, but all are wrapped with Python for simple interfacing.
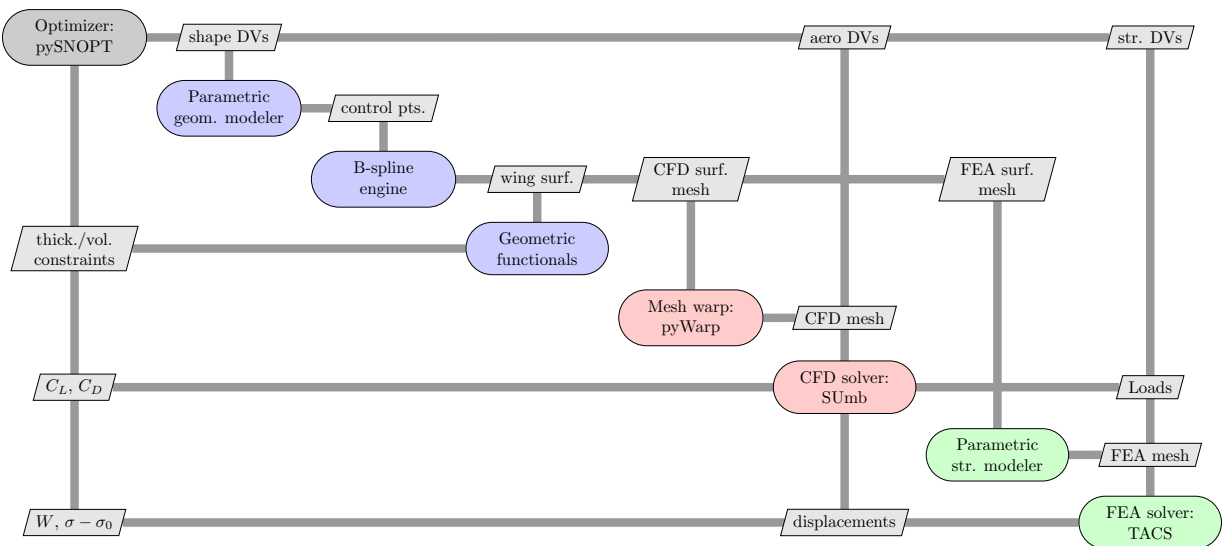

Figure 2: Extended Design Structure Matrix of the software. Blue represents geometry, red represents aerodynamics, and green represents structures.

OPTIMIZER   The optimizer that will be used in this framework is SNOPT [12], wrapped using an optimization framework called pyOpt [13]. SNOPT is a robust optimizer that solves sparse, nonlinear constrained optimization

American Institute of Aeronautics and Astronautics

problems using an active-set reduced-Hessian sequential quadratic programming (SQP) algorithm. The python-based pyOpt framework provides a common interface to a suite of optimizers including SNOPT, enabling access to multiple optimizers with a single optimization problem definition. This project uses pyOptSparse, a modification of the pyOpt framework which enables the specification of constraint Jacobians as sparse matrices and facilitates the construction of gradients and Jacobians using variable names.

GEOMETRY   The parametric geometry modeler is the subject of Sec. III. It parametrizes the shape of the aircraft OML by mapping user-defined parameters and design variables to B-spline control points that define the OML. The B-spline engine maps the control points to multiple discretizations via a sparse matrix—a highly detailed mesh for visualization, the aerodynamic surface mesh, or the structural surface points. The geometry functionals include thickness and volume functions for wings to be used as geometric constraints for the optimization problem.

AERODYNAMICS   For CFD, the Stanford University multi-block (SUmb) flow solver is used here, which has a finite-volume discretization. SUmb can solve the Euler equations and the Reynolds-averaged Navier-Stokes (RANS) equations, but only Euler simulation results are shown in this paper because of the cost of RANS. SUmb uses multigrid during startup and during the 4th order Runge-Kutta time integration stage, after which it switches to a Jacobian-free Newton-Krylov solver for faster convergence. The mesh movement algorithm is called pyWarp, and it can algebraically warp the mesh in response to geometry changes or using a hybrid algebraic-elasticity-based warping, which is the option used in this paper.

STRUCTURES   The parametric structural modeler is described in detail in Sec. IV. Based on inputs from the user, the structural modeler generates Jacobians mapping the B-spline control points of the geometry to the structural surface points and then to the full structural mesh. The FEA solver uses here is the toolkit for the analysis of composite structures (TACS) [14], which uses first-order shear deformation theory (FSDT) quadrilateral shell elements. The aerodynamics and structures solvers used here are detailed in Kenway et al. [15]

The parametric geometry modeler and parametric structural modeler are part of the open-source geometry-centric MDO of aircraft configurations with high fidelity (GeoMACH) tool suite. The objectives and implementation of the software in GeoMACH is presented in Hwang and Martins [8], but both the parametric geometry modeler and the parametric structural modeler have been overhauled and the improvements are detailed in Secs. III and IV.

C.   Usage

One of the high-level objectives listed in Sec. I was a fast turnaround time from geometry definition to high-fidelity MDO. Figure 3 lists the steps involved, with the manual steps in red and automatic steps in blue.
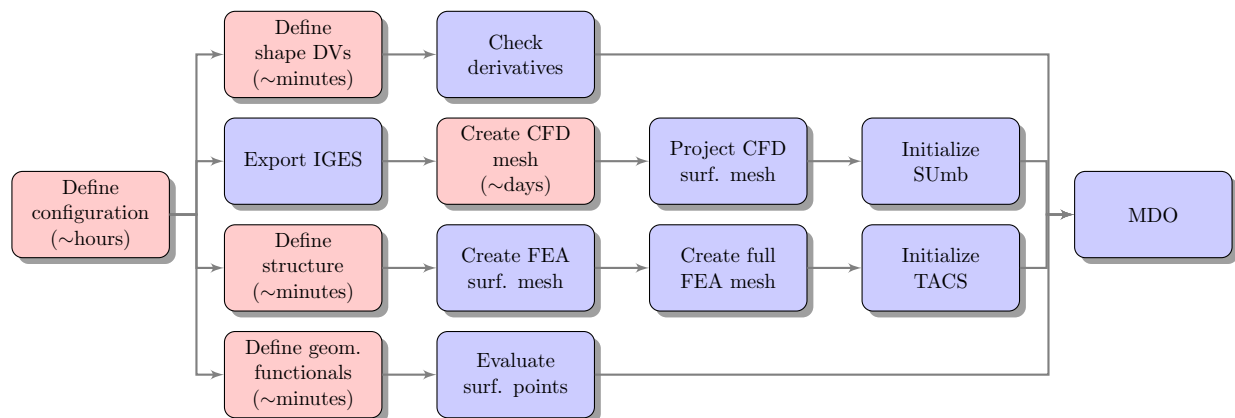

Figure 3: Sequence of steps required to initialize an MDO case. Blue represents automatic steps and red represents manual steps.

Creating a configuration involves defining the topology, which involves specifying the components and how they are connected to each other, and choosing the parameters that define the shape and dimensions of the aircraft. Defining the design variables and their derivatives, takes on the order of minutes unless there are complex design variables that nonlinearly control multiple parameters. The user is required to provide derivatives of the parameters with respect to

their design variables, but this is greatly facilitated by automatic derivative checking. The next step, creating a multi-block CFD mesh, is the most time-consuming step—it takes at least a day even for an experienced user of GUI-based mesh creation software. Another major step is defining the structure using a high-level Python script, after which all remaining steps are automatically executed by the structural modeler. Given these steps, there are three solutions that address the requirement of a short turnaround time and make it possible to have a streamlined design process with high-fidelity multidisciplinary analysis and optimization in the loop.

CFD MESH GENERATION—SOLUTION: CONTINUOUS GEOMETRY DEFORMATION   The multi-block CFD mesh generation step is clearly the bottleneck in the process shown in Fig. 3, but there is a solution that satisfies the requirement of a short turnaround time. Configuration and CFD mesh creation need not happen frequently as baselines for several configurations can be created as an initial step, and the baseline geometry and mesh can be warped for each use. This is enabled by the robustness of the mesh warping algorithm and the fact that the geometry modeler supports continuous deformation for large design changes.

FEA MESH GENERATION—SOLUTION: AUTOMATED UNSTRUCTURED MESHING   Structural mesh generation for detailed airframes can be difficult because the non-regular patches created by intersecting stringers, ribs, spars, etc. and the desired level of isotropy necessitates unstructured meshes. Section IV will describe an automatic unstructured mesh generation algorithm that can generate a global finite element model of the full configuration while handling difficulties such as triangular domains. The algorithm naturally attempts to achieve isotropic elements, which are important for solution accuracy.

HIGH-FIDELITY MODELING—SOLUTION: PARALLEL COMPUTING   The solvers described in the previous section were developed to be as efficient as possible and to scale well with the number of processors. With parallel computing, this enables very fast execution of analyses and optimization; for instance, a 1.5 million cell CFD analysis of the truss-braced wing configuration takes roughly a minute on 64 processors in a high-performance computing (HPC) cluster.

The combination of the above solutions enables a streamlined iterative design process. The bottlenecks of high-fidelity computational design are removed by enabling reuse of CFD meshes, automated FEA mesh generation, and short wall times through highly efficient parallel solvers. Thus, for the majority of the design process, the only manual work involved would be defining different sets of design variables and creating new structural designs, which are steps requiring on the order of minutes, and it would be possible to interactively run analyses and optimizations in a timely manner.

III.   Parametric Geometry Modeler

The design of the parametric geometry modeler is driven by the high-level objectives mentioned in Sec. I: the versatility to model multiple configurations, built-in interfaces to high-fidelity analyses of multiple disciplines, and fitting within a design process with a short turnaround time. As discussed in Sec. C, another key aspect of the parametric geometry modeler is the ability to make large continuous deformations to enable warping existing CFD meshes, where possible, rather than creating a new one.

GEOMETRY REPRESENTATION   To simplify representation of multiple configurations, each is decomposed into components, and an object-oriented approach is adopted as shown in Fig. 4. An aircraft model is an instance of the Configuration class, which inherits from the base Configuration class for all models of that particular layout and topology. A Configuration instance contains Component objects, which fall under two categories: the Primitive class and the Interpolant class. Classes derived from Primitive represent the basic parts of the airframe—lifting surfaces, the fuselage, nacelles, pylons, struts, or others. Classes derived from Interpolant smoothly blend surfaces from Primitive components. Junction instances form the intersection between a Wing instance and another Primitive object, while Tip instances close wing tips and Cone instances are used for the nose cone and tail cone. Each Component instance computes the B-spline control points of its surfaces, which are then aggregated to define a global control point vector that provides a continuous representation of the aircraft OML. The interpolation of Primitive components enables continuous deformation with differentiability always satisfied, and this is one of the main distinguishing features compared to other geometry engines in the literature [16, 17, 18, 19].

GEOMETRY PARAMETRIZATION   The key idea of the geometry parametrization is modeling Primitive components using a concept similar to lofted sections. Since the B-spline control points are arranged in 2-D arrays as shown in Fig. 5, Wing, Body, and Shell components first define 2-D profiles at each section with the origin at the anchor point
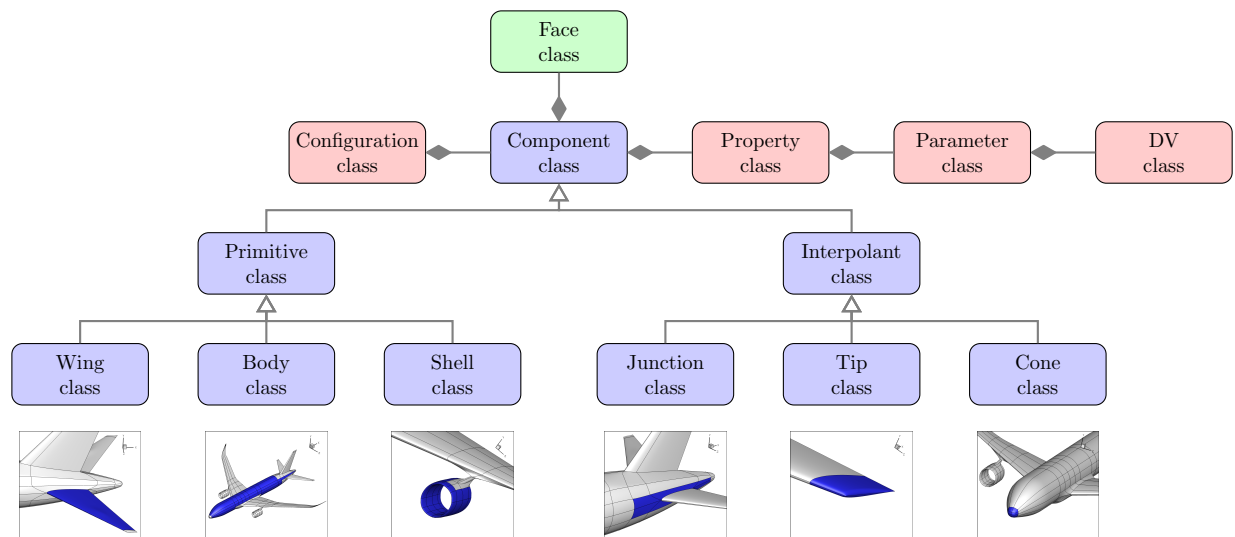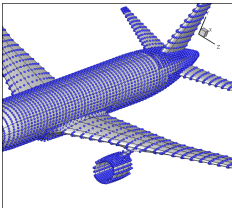
Figure 4: UML diagram of the parametric geometry modeler. Diamonds represent containment and open triangles represent inheritance.

of each section. For each Primitive component, the six *properties* listed in Fig. 5 uniquely define the B-spline control points for the surfaces of interest.

Property	Shape	Description	Usage (e.g. Wing)
position	$(n, 3)$	Anchor point coordinates	Sweep, dihedral, span
rotation	$(n, 3)$	Local to physical frame	Twist, rotated sweep
scale	$(n, 3)$	Stretching in local frame	Chord, thickness
origin	$(n, 3)$	Origin in local frame	
normality	$(n, 3)$	Bool; normal to anchor points	Winglets, vert. stab.
shape	(n_i, n_j)	Shape variables	

Figure 5: List of properties that parametrize Primitive components. In the shape column, n signifies the number of span-wise sections for a wing or the number of stream-wise sections for the fuselage.

The properties are, in turn, parametrized by *parameters* with another B-spline mapping. Thus, when n is large, the number of degrees of freedom can be reduced to a much smaller number while smoothly parametrizing the n sections. This second layer of parametrization offers two advantages. First, this allows the resolution of the geometry representation and manipulation to be independent—the number of span-wise design variables is not fixed to the number of span-wise sections. Second, this parametrization spans the spectrum from high-level aircraft design parameters to local shape variables. If a constant chord is desired, only a single chord value needs to be specified implying a 1st order B-spline, or at the other extreme, n B-spline chord values can be specified. Alternatively, any number in between 1 and n is also possible.

After the B-spline control points belonging to Primitive components are computed, the same is done for the Interpolant components. A wireframe is first computed by interpolating the two components being attached using 3rd degree Bezier curves. The interior of the four-sided domains are then interpolated from the boundary curves as Coons patches. This process is illustrated in Fig. 6.

COMPUTATION OF DERIVATIVES   With a fine surface discretization and thousands of shape design variables, the Jacobian matrix can potentially be very large, necessitating its assembly as a sparse matrix. Computing derivatives as a sparse matrix is made difficult by the fact that there is a sequence of operations connecting the input to the output, and each intermediate quantity is distributed across aircraft components with coupling among them.

Two aspects of the adopted approach greatly facilitate the computation of derivatives. First, for each quantity, the
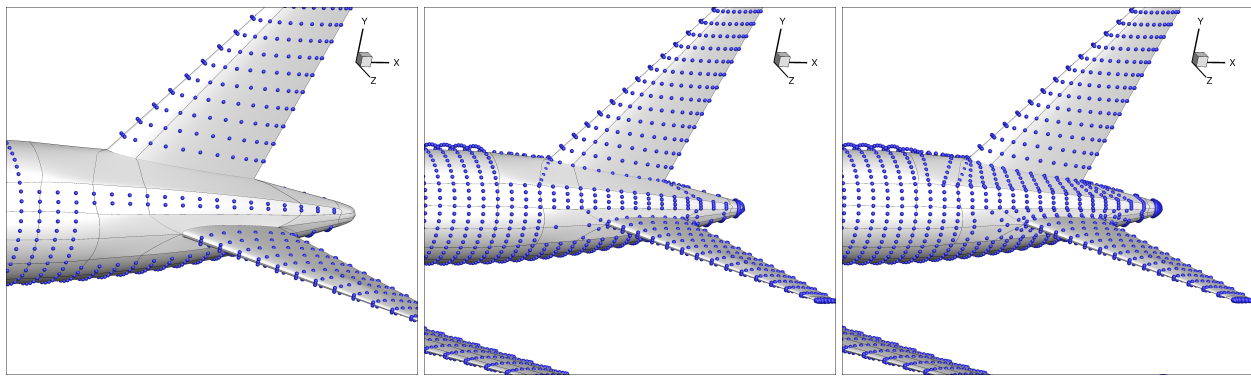
Figure 6: Illustration of the parametrization of a Junction component. Bezier curves are first generated from the intersecting components, and then the interior control points are populated using the formula for a Coons patch.

parts corresponding to each Component are concatenated into a single vector. The parametrization is executed as an explicit sequence, as shown in Fig. 7, and the nonlinearity in the parametrization is contained in only two of the steps. The remaining steps are sparse matrices; therefore, no additional implementation of derivatives is necessary because the Jacobian matrix is the same matrix that defines the mapping itself. Having the sparse Jacobian available also makes it easy to provide the transpose of the matrix, which is needed for the adjoint method.
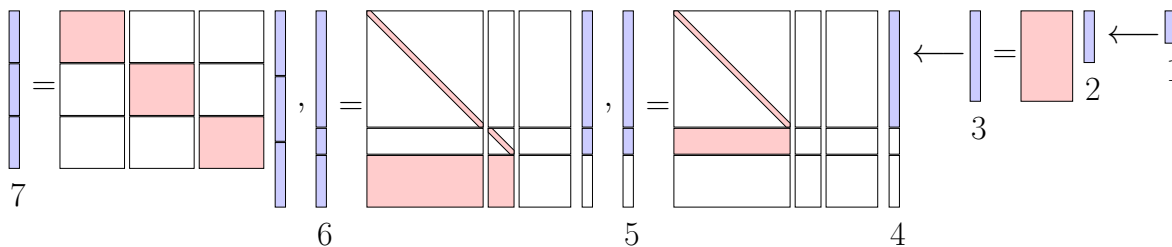

Figure 7: Sequence of operations in the parametrization: (1) Design variables (2) Parameters (3) Properties (4) Primitive control points (5) Primitive and wireframe control points (6) Full control points vector ordered by component and face (7) Unique control points ordered globally.

The second solution addresses storing and accessing the large, concatenated vectors. To simplify global indexing into the concatenated vectors, each quantity in the sequence in Fig. 7 allocates a pair of vectors, one containing the data and one containing the global indices. During initialization, NumPy views are created for sub-vectors of the concatenated vectors, and then reshaped before given to Component and Face instances. This allows Primitive components to work with the control-point arrays in their true, 2-D shapes, instead of a flattened part of a global vector, as shown in Fig 8. Furthermore, if the sparse Jacobian is assembled as a coordinate list, the row and column indices are readily available as in Fig 8.

IV.   Parametric Structural Modeler

The parametric structural modeler has the ability to model detailed airframes including skins, spars, ribs, stringers, frames, longerons, and the cabin floor, among other structural members. The geometry of the internal structure is driven by the OML as the nodal positions are defined in terms of the aircraft OML points, allowing the entire structural mesh to be computed from the B-spline control points of the OML as a linear transformation. This is possible because the internal structure inside wing-type components is embedded in a parametric volume controlled by the upper and lower surfaces of the wing, and likewise, the internal structure inside a component such as a fuselage is projected in a cylindrical volume controlled by the fuselage skin. These internal structures warp, following changes to the fuselage and wing, and because the mapping is linear, updating the full structural mesh takes on the order of only tens of milli-seconds.

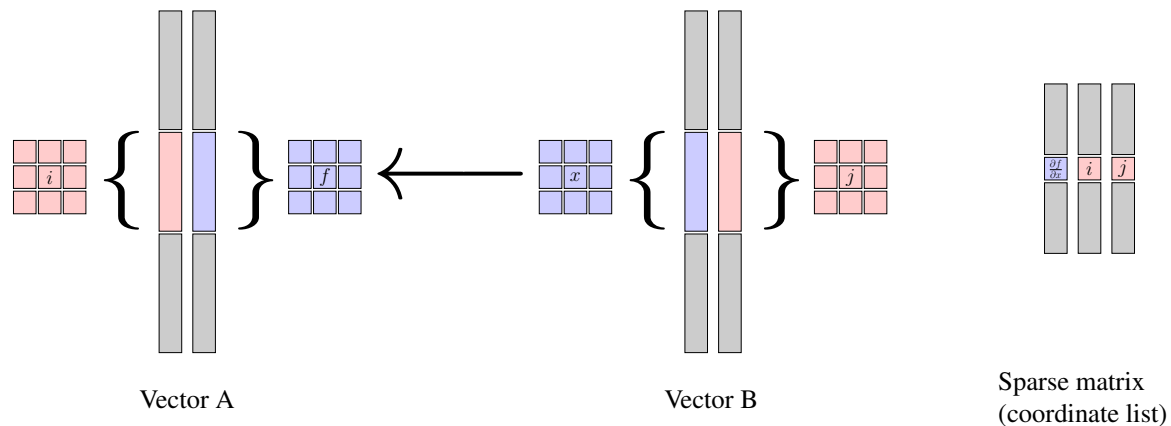American Institute of Aeronautics and Astronautics

Figure 8: Illustration of data storage and accessing. Vectors A and B (e.g. global properties vector and the global parameters vector) are both accessed via reshaped NumPy views onto sub-vectors for each component. Each vector object contains 1-D data array (in blue) and a 1-D indices array (in red) of the same size—this facilitates assembly of the sparse Jacobian.

LINEAR MAPPING   Once the structural layout is computed during initialization, the structural mesh coordinates are defined by a linear mapping from the B-spline control points of the geometry description. Therefore, the derivatives of the structural mesh coordinates can be computed from the known derivatives of the control points with respect to high-level shape design variables in an efficient, accurate, and simple way since the Jacobian is a sparse matrix that does not change. Figure 9 plots derivative contours of the geometry of the OML and structure with respect to a notional design variable.
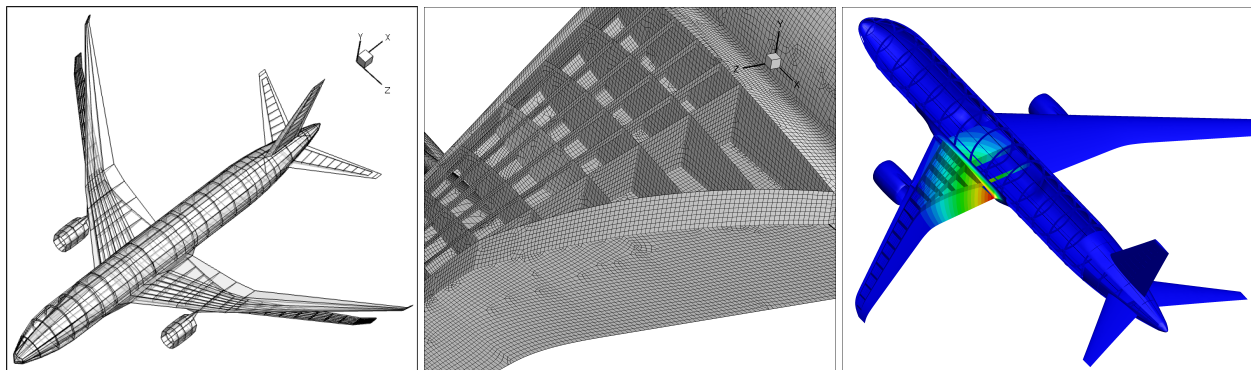

Figure 9: Preview mesh (left), detailed final FEA mesh (center), and derivative contours with respect to a root chord parameter (right). The derivatives get propagated through the wing-fuselage junction and into the centerbody wing box as well.

TWO-STAGE PROCESS   The parametric structural modeler uses a two-stage process that first generates a coarsened preview mesh (shown in Fig 9) for two reasons. First, it provides a quick preview for the user to provide visual feedback on the airframe they have defined. This feature addresses the high-level objective of a fast turnaround time, as the preview mesh takes $\mathcal{O}(\sim \text{sec})$ to compute while the full mesh takes $\mathcal{O}(\sim \text{min})$ for a fine discretization. When the user is interactively designing the structure, the preview mesh is sufficient, so they can make changes and receive feedback in seconds. Furthermore, estimates for the dimensions of the structural members are computed from the preview mesh, and this information is later used to help ensure the quad elements have aspect ratios close to one and angles close to $90°$.

UNSTRUCTURED QUADRILATERAL MESH GENERATION   Automatic computation of a quadrilateral mesh for the entire airframe is challenging for several reasons. The user is allowed to specify any arrangement of structural mem-

American Institute of Aeronautics and Astronautics

bers, and the intersection of spars, ribs, and stringers can create triangular or other non-four-sided patches on the skin that must be meshed with quad elements. Moreover, the meshes for all components must be connected together so an additional span-wise edge created on the wing by a spar must propagate through the wing-body junction and into the fuselage. Another challenge is that features such as taper make it more difficult to have high-quality, isotropic elements since a naive implementation would have the same number of chord-wise elements at the root and at the tip, which creates an imbalance with high resolution at the tip and low resolution at the root of the wing. The structural model for three configurations are shown in Fig. 10.

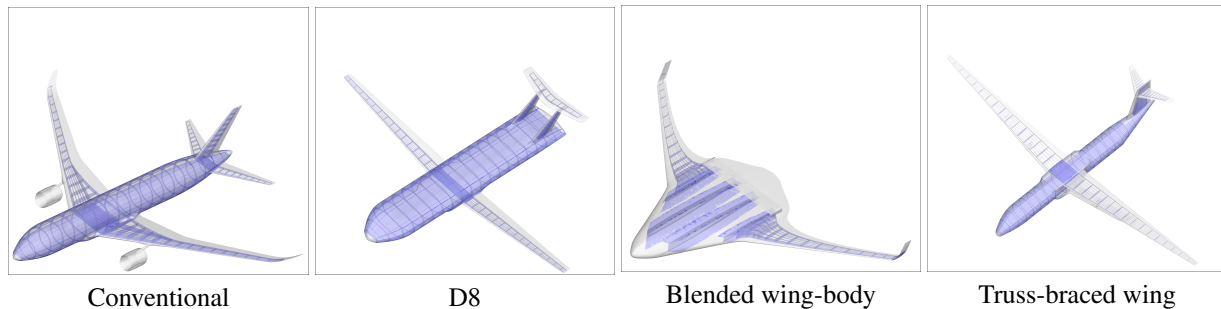| Conventional | D8 | Blended wing-body | Truss-braced wing |

Figure 10: Four configurations with detailed structures.

Two-dimensional quad meshing algorithms fall under three general categories: domain-decomposition [20, 21], advancing-front [22], and triangulation-based methods [23]. The first two—recursively splitting the domain through heuristic algorithms and marching out from boundaries— are unsuited to the current problem because of the line constraints imposed by the structural members intersecting the skin. Two additional ideas that have been successful are topology clean-up [24, 25, 26] and smoothing [27].

There has been work dealing with line constraints in structural mesh generation for marine engineering. Jang et al. use stiffener lines to decompose the domain into regions [28], while Lee et al. use an advancing-front approach on a background triangulation [29]. Park et al. also takes an advancing-front approach, but with topological intersection and clean-up operations [30].

The unique aspect of the current problem is that there are multiple non-planar domains that are connected to each other. The algorithm addresses this by first computing intersection points and discretizing the boundaries of each domain so that each B-spline surface can be quad-patched separately, decoupled from the others. Interior vertices are added, after which a constrained Delaunay triangulation is computed to ensure the intersection lines are respected. A quad-dominant mesh is then produced after ranking all potential merges of adjacent triangles, then a fully quad mesh is produced by splitting each quad and triangle. Finally, the mesh quality is improved using Laplacian smoothing with a second order finite element discretization. This procedure is illustrated in Fig. 11.

V.   Results

As a demonstration of high-fidelity conceptual design optimization, aerodynamic shape optimization was performed on a truss-braced wing design. The geometry, shown in Fig. 12 is based on the Boeing Subsonic Ultra Green Aircraft Research (SUGAR) Volt concept, with a main strut, vertical strut, and a T-tail empennage configuration. A 1.42 million cell multi-block volume mesh was created for Euler CFD analysis and optimization, and a 104,000 degree of freedom mesh was created for structural analysis.

The optimization problem is given in Tab. 1. Lift-constrained drag minimization was performed at a Mach number of 0.74 and c_L of 0.5. Angle of attack was a design variable used to satisfy the lift coefficient constraint, and 531 B-spline shape variables were used to parametrize the upper and lower surfaces of the wing, main strut, vertical strut, and horizontal stabilizer, as well as a portion of the side of the fuselage where the wing and strut attach. Nonlinear thickness constraints were included, although they were not necessary in the design problem because convergence issues forced the addition of the variable bounds that only permitted the control points to thicken the airfoil.

The optimization results are shown in Fig. 13. The initial design had a shock covering almost the entirety of the area bounded by the wing, strut, and fuselage, yielding an initial drag of 810 counts. Through shape optimization, the shock was nearly eliminated and this figure was reduced to 319 counts. The initial design also had a higher lift
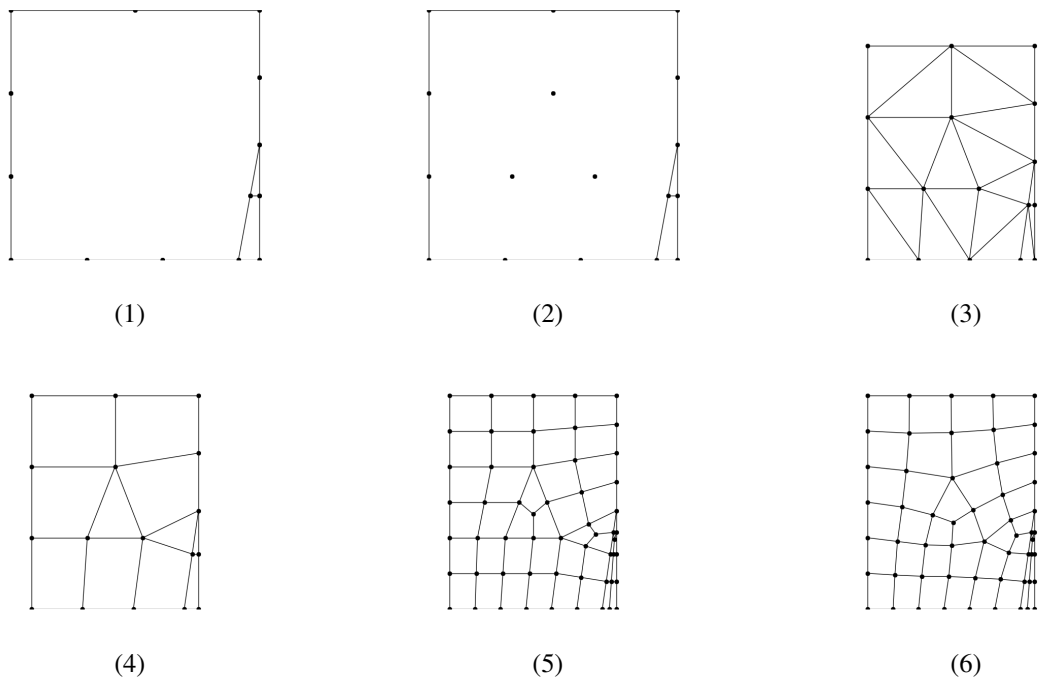
Figure 11: The six steps of the unstructured quad meshing algorithm: (1) Initial domain (2) Interior point insertion (3) Constrained Delaunay triangulation (4) Quad-dominant mesh (5) Fully quad mesh (6) Laplacian smoothing.
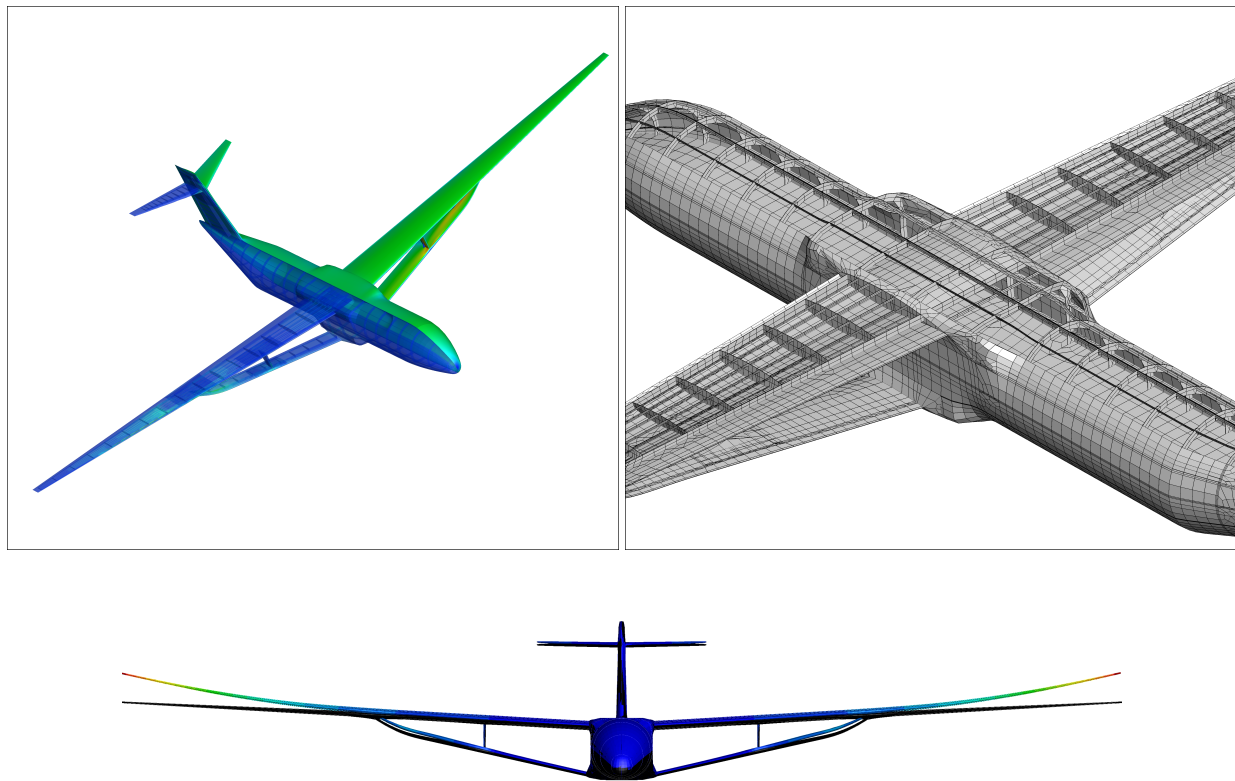

Figure 12: The truss-braced wing design. The structural solutions are the result of uniform loading on the structure.

American Institute of Aeronautics and Astronautics

	Variable/function	Description	Quantity
maximize	c_D	Drag coefficient	
with respect to	α	Angle of attack	1
	$0 \le x_f \le 1$	Fuselage shape variables	5×5
	$0 \le x_{w,U} \le 1$	Wing upper surf. shape variables	10×10
	$0 \le x_{w,L} \le 1$	Wing lower surf. shape variables	10×10
	$0 \le x_{s,U} \le 1$	Strut upper surf. shape variables	8×8
	$0 \le x_{s,L} \le 1$	Strut lower surf. shape variables	8×8
	$0 \le x_{v,U} \le 1$	Vert. strut upper surf. shape variables	5×5
	$0 \le x_{v,L} \le 1$	Vert. strut lower surf. shape variables	5×5
	$0 \le x_{t,U} \le 1$	Tail upper surf. shape variables	8×8
	$0 \le x_{t,L} \le 1$	Tail lower surf. shape variables	8×8
		Total	532
subject to	$c_L - 0.5 = 0$	Lift coefficient constraint	
	$0.25\boldsymbol{t}_w^0 - \boldsymbol{t}_w \le \boldsymbol{0}$	Wing thickness constraints	20×20
	$0.25\boldsymbol{t}_s^0 - \boldsymbol{t}s \le \boldsymbol{0}$	Strut thickness constraints	15×15
	$0.25\boldsymbol{t}_v^0 - \boldsymbol{t}_v \le \boldsymbol{0}$	Vert. strut thickness constraints	10×10
	$0.25\boldsymbol{t}_t^0 - \boldsymbol{t}_t \le \boldsymbol{0}$	Tail thickness constraints	15×15
		Total	951

Table 1: The optimization problem.

coefficient than required; however, even at the prescribed lift coefficient, the initial design had a drag of roughly 750 counts. It can be seen from Fig. 13 that one of the biggest changes is the flattening of the lower surface of the main wing, most likely to avoid the effects of a diverging nozzle. This results in a relatively thick airfoil, but this result is interpreted as a product of solving the Euler equations instead of the RANS equations and the fact that the optimizer did not have the freedom to thin the wing near the quarter-chord mark.

There are two aspects to the significance of these results. The first is the value of the parametric geometry modeler for enabling high-fidelity shape optimization of an unconventional configuration. To eliminate the shock, the optimizer was given fine control of the shape of the fuselage, wing, and struts, which are components that intersect with each other and must be smoothly blended with each other. Moreover, an optimization problem was solved with the intersection point of the strut and wing allowed to move span-wise, but this optimization problem proved to be insensitive to this design variable. This shows that it is possible to perform high-fidelity optimization with high-level design variables that make large geometry changes. The second aspect is the fast turnaround time for computational design demonstrated in this problem. After the initial time investment to create the grid, setting up new design variables and variants of the optimization problem took only on the order of minutes. Furthermore, on 128 processors, optimizations problems of this size required only a few hours to achieve the majority of convergence.

VI.   Conclusion

The motivation for this paper was to develop the methods and algorithms to enable high-fidelity MDO in a conceptual design setting. In Sec. C, three bottlenecks were described: CFD mesh generation, FEA mesh generation, and the execution of high-fidelity tools. The first bottleneck was addressed through a parametric geometry modeler that supports large-scale deformations in a continuous way. With a robust mesh warping algorithm, this enables generating a small number of CFD meshes and warping them to fit a potentially large number of geometries to analyze and optimize. The second was addressed through a fully automated parametric structural modeler that eliminates all manual effort and only requires input from the user to describe the number and layout of structural members to generate full-configuration FEA meshes. In Sec. V, high-fidelity aerodynamic shape optimization of a truss-braced wing concept was demonstrated, showing that optimization of an unconventional configuration can be set up and run in a few hours to quickly obtain a good design with respect to a given objective.

In the process of solving this problem, two contributions were made that have value beyond this project and topic. First, the solution developed for elegantly computing the sparse Jacobian of the parametric geometry modeler can

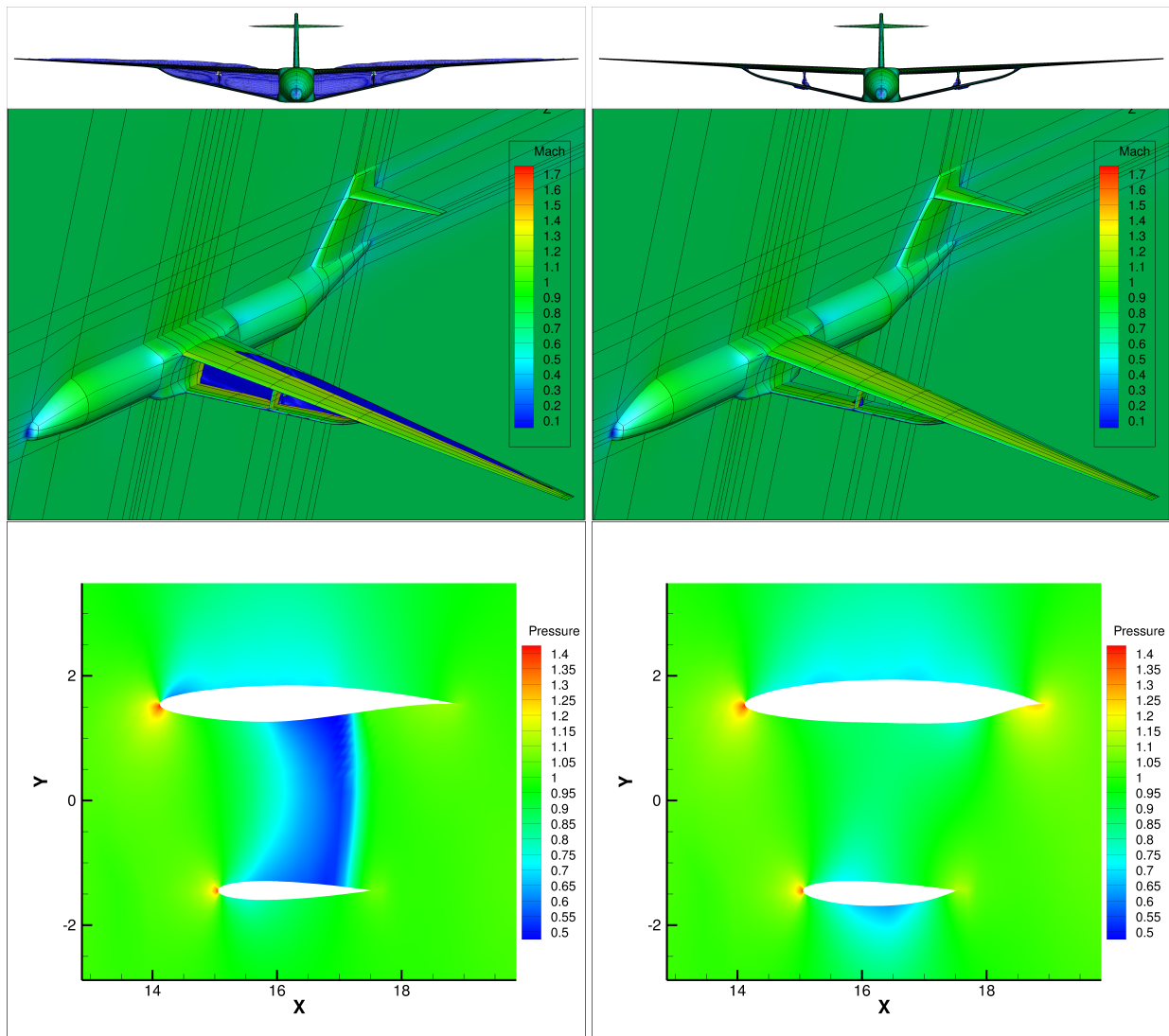American Institute of Aeronautics and Astronautics

Figure 13: Initial (left) and optimized (right) shape and CFD solution of the truss-braced wing configuration. The pressure contours represent a slice near the root of the wing.

American Institute of Aeronautics and Astronautics

apply to any computational modeling setting. The challenge was to compute the derivatives of a component with a sequence of intermediate quantities (design variables to parameters to properties to control points) where each quantity is distributed across multiple classes (wing, fuselage, tail, etc.). This problem was solving in a general way by defining for each quantity an abstract vector class which concatenates the parts from each class, and stores two attributes: the data and the global indices. Each class is given a pointer to a reshaped view of the sub-vector corresponding to the class, for both the data and index arrays. It is explained in Sec. III how this simplifies computing the global sparse Jacobian.

Another general contribution is the unstructured quad meshing algorithm used by the parametric structural modeler. It uses the concept of a preview mesh that provides quick visual feedback for the user on the structural model they defined in the script and also estimates element lengths to improve mesh quality. For the actual mesh generation, the algorithm decomposes the problem according to the B-spline surfaces of the geometry and uses a simple point insertion rule, constrained Delaunay triangulation, and Laplacian smoothing to generate the full FEA mesh.

VII.   Acknowledgments

American Institute of Aeronautics and Astronautics

References

[1] Rutherford, D. and Zeinali, M., "Efficiency Trends for New Commercial Jet Aircraft 1960 to 2008," The International Council on Clean Transportation, 2009.

[2] ICAO Secretariat, "Aircraft technology improvements," *ICAO Environmental Report 2010*, International Civil Aviation Organization, 2010.

[3] Raymer, D. P., *Aircraft Design: A Conceptual Approach*, AIAA, 5th ed., 2012.

[4] Kenway, G. K. W. and Martins, J. R. R. A., "Multi-Point High-Fidelity Aerostructural Optimization of a Transport Aircraft Configuration," *Journal of Aircraft*, Vol. 51, No. 1, January 2014, pp. 144–160. doi:10.2514/1.C032150.

[5] Lyu, Z. and Martins, J. R. R. A., "Aerodynamic Design Optimization Studies of a Blended-Wing-Body Aircraft," *Journal of Aircraft*, 2014. doi:10.2514/1.C032491, (In press).

[6] Hwang, J. T., Lee, D. Y., Cutler, J. W., and Martins, J. R. R. A., "Large-Scale MDO of a Small Satellite using a Novel Framework for the Solution of Coupled Systems and their Derivatives," *Proceedings of the 54th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, Boston, MA, April 2013. doi:10.2514/6.2013-1599.

[7] Martins, J. R. R. A. and Hwang, J. T., "Review and Unification of Methods for Computing Derivatives of Multidisciplinary Computational Models," *AIAA Journal*, Vol. 51, No. 11, November 2013, pp. 2582–2599. doi:10.2514/1.J052184.

[8] Hwang, J. T. and Martins, J. R. R. A., "GeoMACH: Geometry-centric MDAO of Aircraft Configurations with High Fidelity," *Proceedings of the 14th AIAA/ISSMO Multidisciplinary Analysis Optimization Conference*, Indianapolis, IN, Sept. 2012.

[9] Kenway, G. K., Kennedy, G. J., and Martins, J. R. R. A., "A CAD-Free Approach to High-Fidelity Aerostructural Optimization," *Proceedings of the 13th AIAA/ISSMO Multidisciplinary Analysis Optimization Conference*, Fort Worth, TX, Sept. 2010, AIAA 2010-9231.

[10] Gagnon, H. and Zingg, D. W., "Two-Level Free-Form Deformation for High-Fidelity Aerodynamic Shape Optimization," *Proceedings of the 14th AIAA/ISSMO Multidisciplinary Analysis Optimization Conference*, Indianapolis, IN, Sept. 2012.

[11] Lambe, A. B. and Martins, J. R. R. A., "Extensions to the Design Structure Matrix for the Description of Multidisciplinary Design, Analysis, and Optimization Processes," *Structural and Multidisciplinary Optimization*, Vol. 46, August 2012, pp. 273–284. doi:10.1007/s00158-012-0763-y.

[12] Gill, P., Murray, W., and Saunders, M., "SNOPT: An SQP Algorithm for Large-Scale Constraint Optimization," *SIAM Journal of Optimization*, Vol. 12, No. 4, 2002, pp. 979–1006.

[13] Perez, R. E., Jansen, P. W., and Martins, J. R. R. A., "pyOpt: a Python-Based Object-Oriented Framework for Nonlinear Constrained Optimization," *Structural and Multidisciplinary Optimization*, Vol. 45, No. 1, January 2012, pp. 101–118. doi:10.1007/s00158-011-0666-3.

[14] Kennedy, G. J. and Martins, J. R. R. A., "A Parallel Finite-Element Framework for Large-Scale Gradient-Based Design Optimization of High-Performance Structures," *Finite Elements in Analysis and Design*, Vol. 87, September 2014, pp. 56–73. doi:10.1016/j.finel.2014.04.011.

[15] Kenway, G. K. W., Kennedy, G. J., and Martins, J. R. R. A., "Scalable Parallel Approach for High-Fidelity Steady-State Aeroelastic Analysis and Derivative Computations," *AIAA Journal*, Vol. 52, No. 5, May 2014, pp. 935–951. doi:10.2514/1.J052255.

[16] Hahn, A. S., "Vehicle Sketch Pad: A Parametric Geometry Modeler for Conceptual Aircraft Design," *Proceedings of the 48th AIAA Aerospace Sciences Meeting*, Orlando, FL, Jan. 2010.

[17] Rodriguez, D. L. and Sturdza, P., "A Rapid Geometry Engine for Preliminary Aircraft Design," *Proceedings of the 44th AIAA Aerospace Sciences Meeting*, Reno, NV, January 2006, AIAA-2006-0929.

[18] LLC, A., "AVID PAGE," http://www.avidaerospace.com/software/avid-page, Accessed July 11, 2012.

[19] Risse, K., Anton, E., Lammering, T., Franz, K., and Hoernschemeyer, R., "An Integrated Environment for Preliminary Aircraft Design and Optimization," *Proceedings of the 53rd AIAA Structures, Structural Dynamics and Materials Conference*, Honolulu, HI, April 2012, AIAA-2012-1675.

[20] Talbert, J. A. and Parkinson, A. R., "Development of an automatic, two-dimensional finite element mesh generator using quadrilateral elements and Bezier curve boundary definition," *International Journal for Numerical Methods in Engineering*, Vol. 29, No. 7, 1990, pp. 1551–1567. doi:10.1002/nme.1620290712.

[21] Tam, T. and Armstrong, C., "2D finite element mesh generation by medial axis subdivision," *Advances in Engineering Software and Workstations*, Vol. 13, No. 56, 1991, pp. 313 – 324. doi:http://dx.doi.org/10.1016/0961-3552(91)90035-3.

[22] Blacker, T. D. and Stephenson, M. B., "Paving: A new approach to automated quadrilateral mesh generation," *International Journal for Numerical Methods in Engineering*, Vol. 32, No. 4, 1991, pp. 811–847. doi:10.1002/nme.1620320410.

[23] Owen, S. J., Staten, M. L., Canann, S. A., and Saigal, S., "Q-Morph: An indirect approach to advancing front quad meshing," *International Journal for Numerical Methods in Engineering*, Vol. 44, 1999, pp. 1317–1340.

[24] Bunin, G., "Non-Local Topological Clean-Up," *Proceedings of the 15th International Meshing Roundtable*, Springer Berlin Heidelberg, 2006, pp. 3–20. doi:10.1007/978-3-540-34958-7_1.

American Institute of Aeronautics and Astronautics

[25] Canann, S., Muthukrishnan, S., and Phillips, R., "Topological improvement procedures for quadrilateral finite element meshes," *Engineering with Computers*, Vol. 14, No. 2, 1998, pp. 168–177. doi:10.1007/BF01213591.

[26] Verma, C. and Tautges, T., "Jaal: Engineering a High Quality All-Quadrilateral Mesh Generator," *Proceedings of the 20th International Meshing Roundtable*, edited by W. Quadros, Springer Berlin Heidelberg, 2012, pp. 511–530. doi:10.1007/978-3-642-24734-7_28.

[27] Xu, H. and Newman, T., "2D FE Quad Mesh Smoothing via Angle-Based Optimization," *Computational Science ICCS 2005*, edited by V. Sunderam, G. Albada, P. Sloot, and J. Dongarra, Vol. 3514 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, 2005, pp. 9–16. doi:10.1007/11428831_2.

[28] Jang, B.-S., Suh, Y.-S., Kim, E.-K., and Lee, T.-H., "Automatic {FE} modeler using stiffener-based mesh generation algorithm for ship structural analysis," *Marine Structures*, Vol. 21, No. 23, 2008, pp. 294 – 325. doi:http://dx.doi.org/10.1016/j.marstruc.2007.08.001.

[29] Lee, K.-Y., Kim, I.-I., Cho, D.-Y., and wan Kim, T., "An algorithm for automatic 2D quadrilateral mesh generation with line constraints," *Computer-Aided Design*, Vol. 35, No. 12, 2003, pp. 1055 – 1068. doi:http://dx.doi.org/10.1016/S0010-4485(02)00145-8.

[30] Park, C., Noh, J.-S., Jang, I.-S., and Kang, J. M., "A new automated scheme of quadrilateral mesh generation for randomly distributed line constraints," *Computer-Aided Design*, Vol. 39, No. 4, 2007, pp. 258 – 267. doi:http://dx.doi.org/10.1016/j.cad.2006.12.002.