Please cite this document as:
J. T. Hwang and J. R. R. A. Martins. An unstructured quadrilateral mesh generation algorithm for aircraft structures. *Aerospace Science and Technology*, 59:172182, 2016. doi:10.1016/j.ast.2016.10.010.
This document can be found at: `http://mdolab.engin.umich.edu`.

An unstructured quadrilateral mesh generation algorithm for aircraft structures

John T. Hwang[1] and Joaquim R. R. A. Martins[1]

Abstract Because commercial aircraft are built with thin-walled structures, their structural performance is well-modeled using shell-element meshes. However, creating these meshes for the full aircraft configuration can be challenging and presents a bottleneck in the design process, especially in a configuration-level design space. This paper presents an algorithm that automatically creates unstructured quadrilateral meshes for the full airframe based on just the description of the desired structural members. The approach consists in representing each node in the mesh as a linear combination of points on the geometry so that the structural mesh morphs as the geometry changes, as it would, for example, in aerostructural optimization. The algorithm divides the aircraft skin into 4-sided domains based on the underlying B-spline representation of the geometry. It meshes each domain independently using an algorithm based on constrained Delaunay triangulation, triangle merging and splitting to obtain a quadrilateral mesh, and elliptical smoothing. Examples of full-configuration structural meshes are provided, and a mesh convergence study is performed to show that element quality can be maintained as the structural mesh is refined. The algorithm is available as part of the open-source aircraft geometry tool suite, GeoMACH.

Keywords: aircraft design, structural design, mesh generation, aerostructural optimization, structural optimization, elliptical smoothing, finite-element analysis

1 Introduction

The commercial aviation industry faces a pressing need to find ways to reduce aircraft fuel burn given the continued growth of air traffic [1], and rising environmental concerns. This has led to research into new aircraft configurations that deviate significantly from the cylindrical tube-and-wing design that has been used for over half a century, with the hopes of achieving revolutionary breakthroughs in fuel efficiency and in other metrics of interest such as noise. Since there is a lack of knowledge and data on unconventional configurations, there is a need for higher fidelity computational models that can be deployed quickly.

Current aircraft design processes, however, do not take full advantage of computational design tools. Early on in the design process, high-level design decisions are made with the help of relatively low-fidelity and low-accuracy models. This is because high-fidelity models are not well-suited to handle the range of designs considered in conceptual design. As the high-level aspects of the design become frozen—e.g., the placement of the engines—higher-fidelity models are gradually introduced to resolve the finer design parameters—e.g., the shape of structural ribs in the wing. This approach

[1]Department of Aerospace Engineering, University of Michigan, Ann Arbor, MI 48109 (hwangjt@umich.edu, jrram@umich.edu).

is not ideal, because the lowest-accuracy models are used earlier in the design process when the design decisions are the most important. Therefore, it is beneficial to bring higher-fidelity models earlier in the aircraft design process without sacrificing automation, usability, and computational time.

One area in which high-fidelity computational tools can make an impact is the design of the airframe—i.e., the structure of the aircraft. In airframe design, the dominant considerations are the aerodynamic shape and structural layout, which are intrinsically coupled. As an example, thinner and longer wings are beneficial for aerodynamics because they have lower drag, but they also result in more structural weight per unit of wing area due to the higher bending stresses they must withstand. Moreover, with these thin and flexible wings, the aerodynamic loads that produce lift cause the wing to twist, which in turn causes higher aerodynamic loads, so there is a strong coupling between these disciplines.

High-fidelity aerostructural optimization addresses this coupling by simultaneously optimizing the aerodynamic shape of the airframe and the sizing of the structural members [2, 3]. This leads to at least $\mathcal{O}(100)$ design variables that must be optimized. There can also be thousands or more structural failure constraints, but these can be reduced to a single or a small number using constraint aggregation methods, such as the Kreisselmeier–Steinhauser functional [4, 5]. At this scale, gradient-based optimization is the only feasible approach [6, 7], especially given the large computational cost of a structural or aerostructural simulation. For derivative computation, the adjoint method is the best choice in terms of efficiency for most problems because it computes all derivatives at a computational cost nearly independent of the number of design variables.

For high-fidelity structural modeling and design, which is the focus of this paper, the common approach is to use a shell-element model. Aircraft are well-represented by shell elements because the structural members in aircraft wings and fuselages are in general very thin due to the premium placed on weight reduction. Using these shell elements, it is possible to model the ribs, stringers, spars, and stiffeners in the wings, as well as floor beams, frames, longerons, and bulkheads in the fuselage, and the shell elements carry bending, twisting, axial, and shear loads.

One of the bottlenecks in airframe structural analysis is the creation of the structural mesh, which typically requires extensive manual effort and a high level of expertise. Meshing tools aim to alleviate this bottleneck. Given the overarching motivation for this paper—the rapid design and evaluation of unconventional aircraft configurations—there are four requirements for such a structural meshing tool. First, the structural mesh should be global to enable quantitative trade studies comparing configurations; that is, the mesh should model the full configuration (not just the wing), and there should not be separate, disconnected meshes for different aircraft components. Second, the mesh generation should be automatic so that given a description of the desired structural members, number and location of ribs, placement of spars, *etc.*, the mesh should be created without any additional manual effort. Third, the mesh should be computed as a function of shape design variables because for aerostructural design and optimization, shape changes must automatically morph the structural mesh. Fourth, the mesh definition should be differentiable, and the computation of the derivatives of the structural mesh coordinates with respect to the shape design variables should be efficient. Existing aircraft structural meshing tools do not satisfy all four requirements; some mesh only the wing [8], some lack automation because they use an external tool to generate an unstructured quad mesh [9], and others do not compute the mesh as a differentiable function of shape changes [10].

There are additional application-specific requirements that we have not addressed because our primary focus is on the four fundamental requirements listed above. One such requirement is to be able to handle the parametrization of the composite layup [11, 12]. The modeling of the composite layup in each element is something that can be handled by a separate tool that takes the generated

structural mesh as an input. Another requirement is multidisciplinary data transfer, especially the transfer of displacements and loads to and from computational fluid dynamics (CFD) analysis. However, it is possible to use a general load and displacement transfer algorithm that is independent of the structural mesh [13, 2]. Another application-specific requirement is the modeling of control surface deflections, which is limited when using structured multi-block CFD, but is possible when approximating the control surfaces as morphing surfaces with a continuous trailing edge [14].

In this paper, we present an automatic unstructured quadrilateral mesh generation algorithm for aircraft structures that uniquely satisfies the four requirements mentioned above. The algorithms starts with a B-spline surface geometry representation and a list of requested structural members defined in terms of parametric locations on the surfaces. It then splits the geometry into domains, meshes each domain independently using constrained Delaunay triangulation (CDT) as well as merging and splitting operations, and then applies Laplacian smoothing as a final step.

The paper proceeds as follows. We first present the overall approach, discussing the geometry representation details, computation of the structural mesh, and the interface through which the requested structural members are specified. Next, we present the actual mesh generation algorithm, which divides into the global algorithm at the configuration level and a local mesh generation algorithm. Finally, we present results, including aircraft structural meshes created using the proposed algorithm.

2 Approach

In this section, we present the overall approach for the structural mesh computation. More specifically, we describe the assumed form of the geometry representation for the aircraft outer mold line (OML), explain how the structural nodes are computed from the geometry using a linear map, and then discuss how a user would specify the desired structural members.

For the results in this paper, we use the geometry-centric MDO of aircraft configurations with high fidelity (GeoMACH) tool suite [15, 16]. GeoMACH is an open-source software library that models aircraft geometries using a patchwork of untrimmed B-spline surfaces, and includes an aircraft parametrization to support high-fidelity aircraft shape design optimization. The structural mesh generation algorithm developed here is part of GeoMACH, which is available through an open source license[1].

2.1 Geometry representation

The only requirements on the OML geometry representation are that it is continuous and watertight. As mentioned previously, we use the geometry modeler in the GeoMACH tool suite for the figures presented in this paper. GeoMACH represents the geometry using untrimmed B-spline surfaces, though this is not the only choice with which the structural mesh generation algorithm would work. B-splines are piecewise polynomials used frequently in computer-aided design because of their favorable mathematical properties: compact support for a desired order and smoothness, and flexibility in terms of the number of control points and polynomial degree. B-spline surfaces are tensor products of B-spline curves that maintain the advantages of smoothness and sparsity. Figure 1(a) illustrates how a conventional wing-body-tail aircraft geometry can be constructed with 4-sided B-spline surfaces.

An important feature of the geometry modeler is the ability to perform point-to-surface projections. These are required so that we can evaluate surface nodes for modeling the aircraft skin and for interpolating interior nodes. For the projections, the B-spline implementation in GeoMACH performs a Newton search to find the parametric coordinates (u, v) on the surface that yield the closest point to the given point that we are projecting. Since this procedure can fail in some cases,

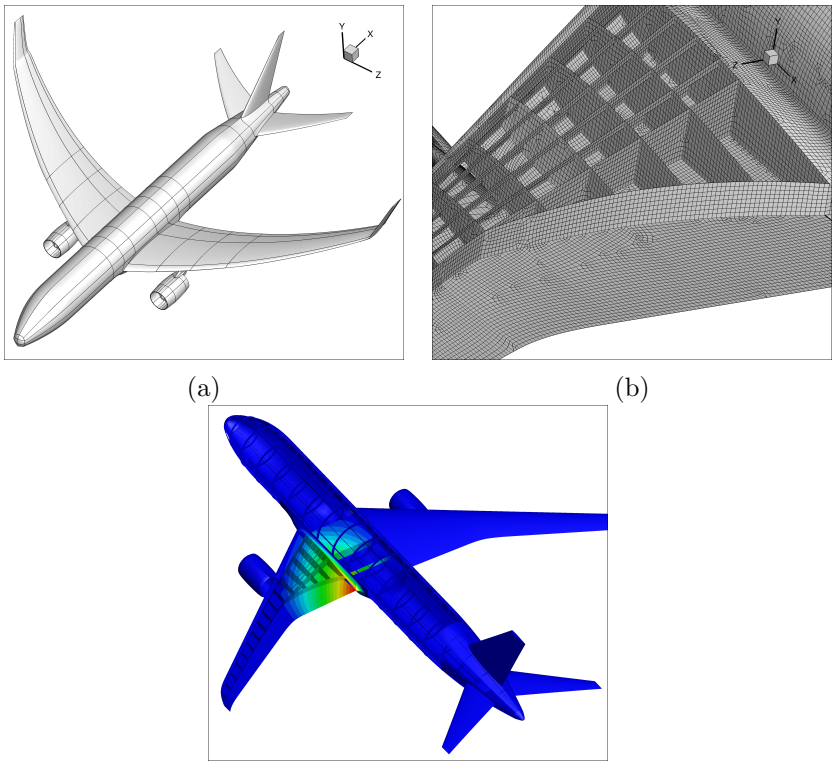


Figure 1: Conventional configuration geometry (a), final structural mesh (b), and derivative contours of nodal coordinates with respect to a root chord design variable (c).

the algorithm first computes a brute-force closest-point search on a structured discretization of the surface, to be used as the initial point for the Newton search and as an alternative in case the Newton search does not converge. To handle cases in which the closest point is at one of the surface edges, there are provisions during the iteration loop to detect such a case and exit. This projection algorithm occurs at the individual surface level, so in general, all the B-spline surfaces that make up a geometry must be searched to find the closest point on the geometry to a given point. However, in the current application, only the small number of B-spline surfaces that comprise the aircraft component of interest (e.g., the upper surface of the wing) must be searched, since the relevant component is known for a given structural node.

2.2 Structural mesh morphing

Given the continuous B-spline geometry representation, the approach we take is to define the structural mesh as a linear map from the geometry description. In other words, each node in the structural mesh is computed via a linear combination of points on the OML. Therefore, as the shape design variables change the OML, we only need to apply this linear map to compute the updated structural mesh nodes that reflect the geometry changes.

In the case of GeoMACH, a two-level parametrization is used for the geometry manipulation. First, the shape design variables are mapped to the control points of the B-spline surfaces that define the geometry. Next, the resulting control points are mapped to the discrete evaluation points on the geometry by multiplying the sparse B-spline basis functions. The surface points from the structural mesh simply represent one discretization of the B-spline geometry, and the parametric coordinates of these surface points are computed using the projection algorithm described in Sec. 2.1.

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 with respect to the B-spline control points are given by a Jacobian that is constant and sparse. Using the chain rule, this Jacobian must then be combined with the derivatives of the B-spline control points with respect to the shape design variables to be used for optimization. Figure 1(c) plots derivative contours of the geometry of the OML and structure with respect to a shape design variable, which is the root chord in this case.

2.3 User interface

We now describe how the user specifies their desired structural design. In GeoMACH, the geometry surfaces are already divided by aircraft component, which includes primitive components such as the wing and the fuselage, and junction components for the fairings and blending regions where primitive components intersect. One can define a structural member by specifying the aircraft components and the parametric coordinates on those components from which the structural member is interpolated. As an example, a wing rib is defined by specifying the upper and lower surfaces of a wing component, along with the (u, v) coordinates in the chord-wise and span-wise directions, respectively, where the span-wise parametric coordinate remains constant. The variation in the vertical direction is achieved by varying the weights of the points on the upper and lower wing surfaces, 0 to 1 for one surface and 1 to 0 for the other surface. Therefore, the internal structure inside wing-type components is essentially embedded in a parametric volume controlled by the upper and lower surfaces of the wing. 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 milliseconds. The objective of

the structural mesh generation algorithm described in Sec. 3 is to compute this Jacobian matrix that maps the vector of surface nodes on the B-spline geometry to the vector of structural nodes.

3 Mesh generation algorithm

In this section, we present the structural mesh generation algorithm in three parts: the overall algorithm at the global (aircraft configuration) level, the quadrilateral mesh generation on a single domain, and the elliptical smoothing step.

3.1 Global mesh generation algorithm

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 members (spars, ribs, stringers, *etc.*), and the imprinting of these members onto the surface skin can create difficulties in the meshing process, as shown in Fig. 2. For instance, a spar, rib, and a stringer may form the sides of a triangle on the skin that must eventually be meshed with quad elements. An example of this issue is the orange region in Fig. 2. Moreover, the skins of all the aircraft components must be meshed simultaneously to form a global mesh containing the aircraft wing, fuselage, tail, pylons, and nacelles. Another challenge is that features such as taper make it more difficult to have high-quality, isotropic elements since a regular (structured) mesh would have the same number of chord-wise and through-thickness 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 high-level approach of the global structural mesh generation algorithm is to divide the problem of meshing the skin into multiple smaller, decoupled mesh generation sub-problems. The geometry representation in GeoMACH consists of a union of untrimmed B-spline surfaces, so these surfaces provide a natural subdivision of the skin mesh generation problem. The advantage of this subdivision is that the problem of meshing the complex network of aircraft skin components is simplified to meshing a set of smaller 4-sided domains, which is computationally cheaper. An example of a 4-sided domain defined by a B-spline surface is shown in red in Fig. 2.

For a global mesh, there must be continuity between boundaries between domains—i.e., B-spline surfaces—because an edge that intersects the boundary from one domain must propagate into the neighboring domain to avoid hanging nodes, as shown by the dotted blue curve in Fig. 2. To satisfy this requirement, the algorithm first loops through all the domains and computes the intersections between the imprinted features (edges where ribs and spars intersect the skin), and the bounding edges of the domains. The result of this process is now a set of decoupled mesh generation sub-problems, where each one operates on a 4-sided domain. The shared edges are discretized once so that the resultant global mesh matches at these bounding edges.

On top of this decomposition-based approach, the global structural mesh generation algorithm uses a two-step process that first generates a coarsened "preview mesh". This is done for two reasons. First, it provides a quick preview for the user to visualize the airframe they have defined. This feature has a fast turnaround time as the preview mesh takes $\mathcal{O}(\text{sec})$ to compute, while the full mesh takes $\mathcal{O}(\text{min})$ 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. The second reason is that 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 as close as possible to 1 and angles close to 90°.

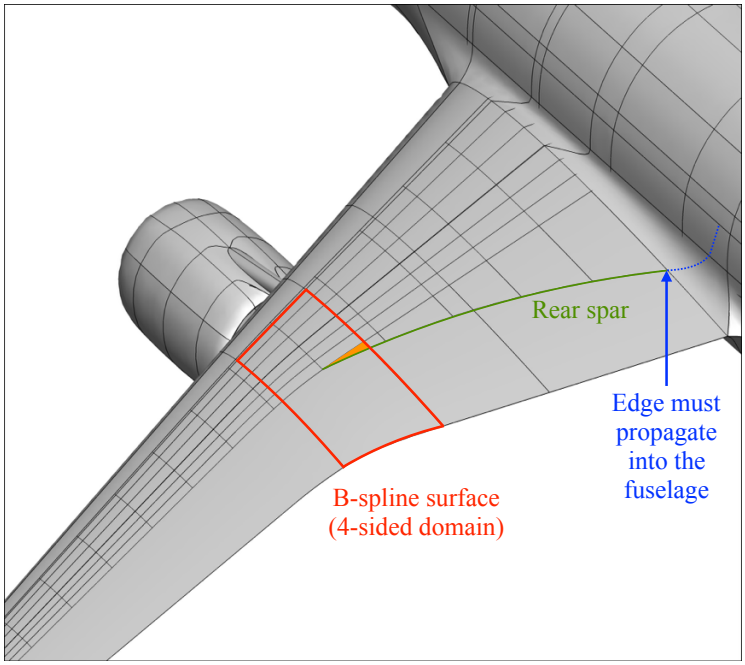


Figure 2: The local mesh generation algorithm is applied separately to each B-spline surface in the geometry definition. The area in orange highlights an example of a triangular region created by the intersection of spars, ribs, and stringers imprinting on the aircraft skin. The dotted blue curve shows an edge that needs to be propagated from one component into an adjacent component.

3.2 Local mesh generation algorithm

Once the aircraft skin is divided into domains, a local mesh generation algorithm turns each domain into an unstructured quadrilateral mesh. As discussed in Sec. 3.1, the consistency of nodes along the boundaries between neighboring domains is assured in the global mesh generation step. However, the main challenge during the local mesh generation step is to make sure that all internal members—ribs, spars, bulkheads, *etc.*—that intersect the skin within a given domain show up in the mesh on the skin and the nodes are consistent. To ensure this, we solve a 2-D quadrilateral mesh generation problem with line constraints.

In general, 2-D quad meshing algorithms fall under three general categories: domain-decomposition [17, 18], advancing-front [19], and triangulation-based methods [20]. The first two—recursively splitting the domain through heuristic algorithms and marching out from boundaries, respectively—are not suitable for 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 [21, 22, 23] and smoothing [24].

There has been work dealing with line constraints in structural mesh generation for marine engineering. Jang et al. [25] use stiffener lines to decompose the domain into regions, while Lee et al. [26] use an advancing-front approach on a background triangulation. Park et al. [27] also uses an advancing-front approach, but with topological intersection and clean-up operations. The unique aspect of the current problem is that there are multiple nonplanar domains that are connected to each other. As mentioned previously, the global mesh generation algorithm addresses this by discretizing the boundaries of each domain so that each one can be meshed separately, decoupled from the others.

The local mesh generation algorithm consists of six stages, as illustrated in Fig. 3. The figure

shows a domain for illustrative purposes, containing a vertical edge extending from the top to the bottom of the domain, two diagonal edges intentionally chosen to form a triangular region, and a shorter edge that is floating by itself near the center of the domain. The six stages are as follows:

1. *Initial domain:* We start with a 4-sided domain representing a single B-spline surface, with the internal members intersecting this surface pre-determined.

2. *Discretization:* We discretize the boundaries and the interior of the domain. The boundaries are simply discretized using a global parameter representing the requested resolution. This guarantees that the bounding edges shared by two neighboring domains always agree on the boundary nodes because all domains use the same resolution parameter. The interior of the domain is populated with a grid of points that are spaced based on the sizes of the element boundaries, as measured from the preview mesh.

3. *Triangulation:* We perform CDT on the domain while respecting the edges from the boundaries and from the intersecting structural members. For our implementation, we use the implementation in TRIPACK [28].

4. *Quad-dominant mesh:* From the triangulation, we obtain a quad-dominant mesh by ranking all potential merges of adjacent triangles based on how close the angles would be to 90°. The triangles are merged according to this ranking until no possible merges remain.

5. *Fully-quad mesh:* We split all quads into four smaller quads and all triangles into three quads using its centroid to obtain a fully quad mesh.

6. *Smoothing:* We perform an elliptical smoothing as the last step, which is explained in more detail in the next section.

3.3 Elliptical smoothing

Here, we provide more detail on the last of the six steps in the local mesh generation algorithm. This step performs elliptical smoothing on the fully quadrilateral mesh produced for each domain.

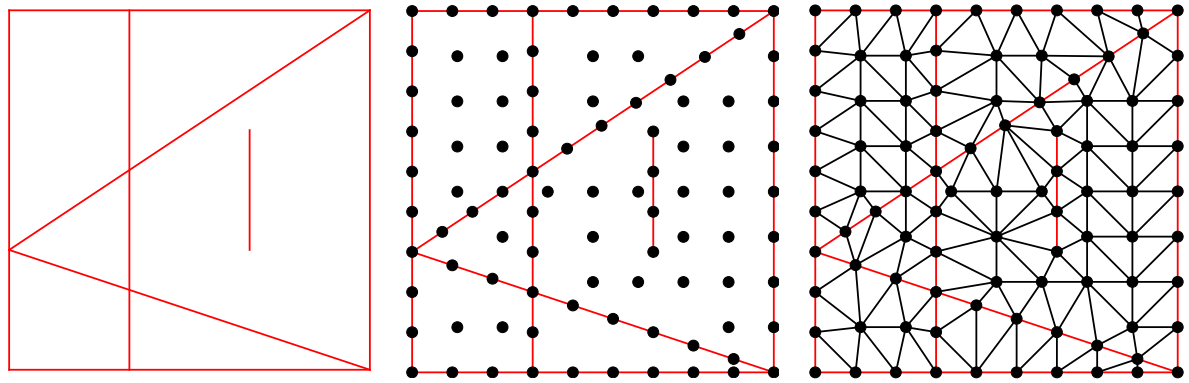
There are several types of general mesh smoothing algorithms. One type solves optimization problems to maximize element quality [29] quantified using a combination of element aspect ratio, angle, and area. However, nonlinear optimization is not always robust and can be inefficient. Another type formulates elliptical partial differential equations (PDEs) with derivatives taken with respect to the physical coordinates of the elements [30], but the resulting nonlinearity causes similar drawbacks to the optimization-based approach.

According to Spekreijse [31, Ch. 4], the first use of the Laplace equations in grid generation was by Winslow [32], who interprets the mesh lines as equipotentials, leading to the equations

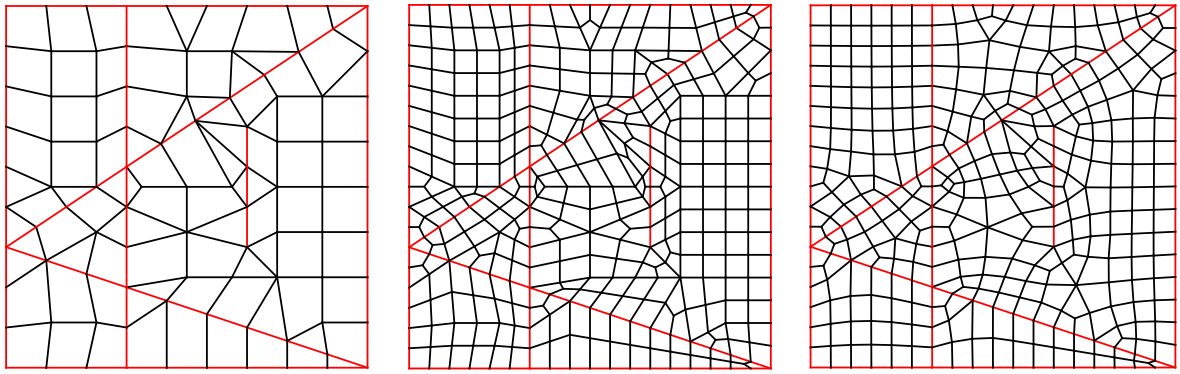
$$\xi_{xx} + \xi_{yy} = 0 \tag{1}$$

$$\eta_{xx} + \eta_{yy} = 0, \tag{2}$$

where $\xi(x, y)$ and $\eta(x, y)$ are potential functions. Since Winslow [32] deals with triangles, a third set of lines is defined by the condition that the three sets of lines intersect at 60° at any point in parametric space. The triangular mesh can be drawn from the equipotential curves of ξ and η, together with the third set of lines. Equations (1) and (2) are solved by inverting them, turning them into nonlinear elliptic equations. These nonlinear equations can be tuned using control functions to control boundary orthogonality and off-wall spacing for applications such as viscous CFD.



(1) Initial domain (2) Discretization (3) Triangulation



(4) Quad-dom. mesh (5) Fully-quad mesh (6) Smoothing

Figure 3: The six steps of the unstructured quad meshing algorithm.

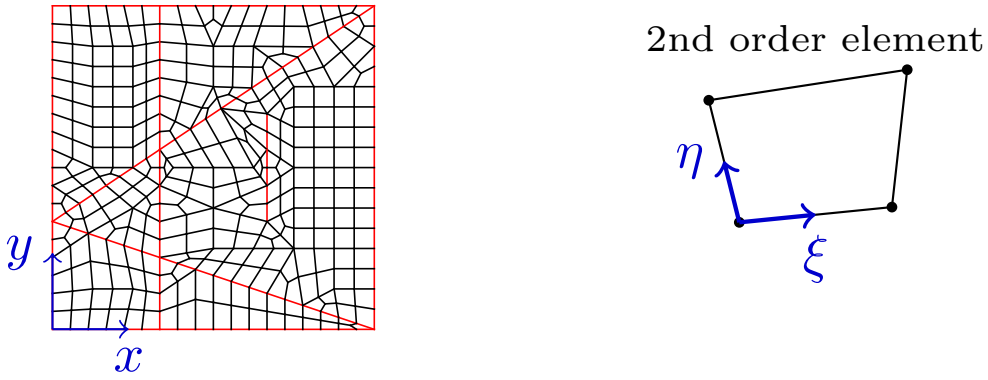


Figure 4: Global (left) and local (right) coordinate frames for the elliptical smoothing.

Far from these boundaries, these nonlinear Poisson equations are well-approximated by the linear Laplace equations,

$$x_{\xi\xi} + x_{\eta\eta} = 0 \tag{3}$$

$$y_{\xi\xi} + y_{\eta\eta} = 0. \tag{4}$$

The approach adopted here is based on these equations. However, we are forced to work with an unstructured quadrilateral mesh, departing from the traditional Laplacian or Poisson-based smoothing, because the imprinted features do not allow for a structured mesh. The parametric coordinates ξ and η are not defined uniformly for the entire mesh; rather, they are defined within each element only.

An alternative approach would be to apply a finite number of Gauss–Seidel or point Jacobi iterations for the linear system given by equations (3) and (4) where the second derivative operator is discretized using the second-order finite-difference formula. However, we use a sparse direct solver instead to solve the linear system fully, rather than achieving only partial convergence because it is very fast computationally—the solution time is negligible compared to the CDT step.

Therefore, our approach is to solve Equations (3) and (4) on each element using a finite-element discretization of the quadrilateral mesh. In this manner, each element benefits from the tendency of the Laplace equation to produce isotropic quadrilaterals with angles close to 90°. However, since neighboring elements share edges and vertices, the global finite-element solution finds the nodal coordinates that find a compromise across all the elements.

In contrast to traditional elliptical smoothing, we solve a linear system rather than nonlinear equations (3) and (4), so we gain efficiency, simplicity, and robustness. The computation time is negligible compared to the CDT step for this reason, and also due to the sparse assembly and solution of the finite-element system.

For simplicity in deriving the equations, we use ϕ in place of x and y. Therefore, the smoothing step computes the x and y coordinates of the nodes in the quad mesh by solving Laplace's equation, in the form,

$$\frac{\partial^2 \phi}{\partial \xi^2} + \frac{\partial^2 \phi}{\partial \eta^2} = 0, \tag{5}$$

where ϕ represents either x or y, and ξ and η are the local coordinates in the element frame, as shown in Fig. 4.

Laplace's equation is solved using a finite-element discretization and the Galerkin method of weighted residuals. Assuming n is the order of the finite-element basis function, the scalar field for

x or y within an element is given by

$$\phi(\xi,\eta) = \sum_{k=1}^{n}\sum_{l=1}^{n} u_{kl}\cdot f_k(\xi)f_l(\eta), \tag{6}$$

where u_{kl} is a nodal value and f_k or f_l represents a basis function. The Galerkin method of weighted residuals is applied to Laplace's equation (5), yielding

$$\int\int_A \left[\frac{\partial^2\phi}{\partial\xi^2} + \frac{\partial^2\phi}{\partial\eta^2}\right] f_i(\xi)f_j(\eta)dA = 0. \tag{7}$$

Prior to inserting the approximate form for ϕ, it is beneficial to apply a step similar to integration by parts to reduce the order of the integrand. Using the product rule, the two terms in the integrand can be written as

$$f_i(\xi)f_j(\eta)\frac{\partial^2\phi}{\partial\xi^2} = \frac{\partial}{\partial\xi}\left(f_i(\xi)f_j(\eta)\frac{\partial\phi}{\partial\xi}\right) - \frac{\partial}{\partial\xi}\left(f_i(\xi)f_j(\eta)\right)\frac{\partial\phi}{\partial\xi} \tag{8}$$

$$f_i(\xi)f_j(\eta)\frac{\partial^2\phi}{\partial\eta^2} = \frac{\partial}{\partial\eta}\left(f_i(\xi)f_j(\eta)\frac{\partial\phi}{\partial\eta}\right) - \frac{\partial}{\partial\eta}\left(f_i(\xi)f_j(\eta)\right)\frac{\partial\phi}{\partial\eta}, \tag{9}$$

and inserting the expressions in Eqs. (8) and (9) into Eq. (7) yields,

$$\begin{aligned}&\int\int_A \left[\frac{\partial}{\partial\xi}\left(f_i(\xi)f_j(\eta)\frac{\partial\phi}{\partial\xi}\right) + \frac{\partial}{\partial\eta}\left(f_i(\xi)f_j(\eta)\frac{\partial\phi}{\partial\eta}\right)\right]dA - \\ &\int\int_A \frac{\partial}{\partial\xi}\left(f_i(\xi)f_j(\eta)\right)\frac{\partial\phi}{\partial\xi}dA - \int\int_A \frac{\partial}{\partial\eta}\left(f_i(\xi)f_j(\eta)\right)\frac{\partial\phi}{\partial\eta}dA = 0.\end{aligned} \tag{10}$$

In the first term of Eq. (10), the integrand is the divergence of a vector field and the integration occurs over a compact subset, so Gauss's theorem can be applied to establish the equality,

$$\int\int_A \nabla\cdot\begin{bmatrix} f_i(\xi)f_j(\eta)\dfrac{\partial\phi}{\partial\xi} \\ f_i(\xi)f_j(\eta)\dfrac{\partial\phi}{\partial\eta}\end{bmatrix} dA = \oint_{\partial A}\begin{bmatrix} f_i(\xi)f_j(\eta)\dfrac{\partial\phi}{\partial\xi} \\ f_i(\xi)f_j(\eta)\dfrac{\partial\phi}{\partial\eta}\end{bmatrix}\cdot\hat{n}\,ds, \tag{11}$$

but $f_i(\xi)$ and $f_j(\eta)$ are zero on ∂A for appropriately chosen shape functions, so the entire line integral on the right-hand side is zero. Setting the divergence term to zero, and inserting the expression for ϕ into Eq. (10) yields,

$$\int\int \frac{\partial}{\partial\xi}\left(f_i(\xi)f_j(\eta)\right)\frac{\partial}{\partial\xi}\left(\sum_{k=1}^{n}\sum_{l=1}^{n} u_{kl}f_k(\xi)f_l(\eta)\right)d\xi d\eta + \tag{12}$$

$$\int\int \frac{\partial}{\partial\eta}\left(f_i(\xi)f_j(\eta)\right)\frac{\partial}{\partial\eta}\left(\sum_{k=1}^{n}\sum_{l=1}^{n} u_{kl}f_k(\xi)f_l(\eta)\right)d\xi d\eta = 0 \tag{13}$$

Rearranging, the final form of the discretized equations is,

$$\sum_{k=1}^{n}\sum_{l=1}^{n}\left[\int f_i'(\xi)f_k'(\xi)d\xi\int f_j(\eta)f_l(\eta)d\eta + \int f_i(\xi)f_k(\xi)d\xi\int f_j'(\eta)f_l'(\eta)d\eta\right]u_{kl} = 0. \tag{14}$$

Introducing the matrices F and F' to simplify, the previous equation becomes

$$\sum_{k=1}^{n}\sum_{l=1}^{n}\left[F'_{ik}F_{jl}+F_{ik}F'_{jl}\right]u_{kl}=0 \tag{15}$$

or

$$\sum_{k=1}^{n}\sum_{l=1}^{n}K_{ij,kl}\,u_{kl}=0, \tag{16}$$

where $K_{ij,kl}$ represents the entry in the global finite element matrix located at the row corresponding to node (i,j) and the column corresponding to node (k,l) in the current element. Similarly, u_{kl} is the value of x or y for the node in the global ordering indexed as (i,j) in the current element.

Though the true basis functions vanish at their boundaries, it is valid to derive element matrices by considering only the parts of basis functions within a given element. A linear equation corresponding to an interior node includes contributions from all the adjacent elements, which together partition the true basis function associated with this node. A linear equation corresponding to a boundary or constrained interior node does not influence the finite-element solution.

The basis functions and the matrices for second-order elements are

$$\begin{array}{l} f_1(t)=1-t \\ f_2(t)=t \end{array} \text{ with } \quad F=\frac{1}{6}\begin{bmatrix}2&1\\1&2\end{bmatrix} \quad \text{and} \quad F'=\begin{bmatrix}1&-1\\-1&1\end{bmatrix} \tag{17}$$

and those for third order-elements are

$$\begin{array}{l} f_1(t)=2t^2-3t+1 \\ f_2(t)=4t-t4^2 \\ f_3(t)=2t^2-t \end{array} \text{ with } \quad F=\frac{1}{30}\begin{bmatrix}4&2&-1\\2&16&2\\-1&2&4\end{bmatrix} \quad \text{and} \quad F'=\frac{1}{3}\begin{bmatrix}7&-8&1\\-8&16&-8\\1&-8&7\end{bmatrix}. \tag{18}$$

Having derived the local element matrices, the global finite element equations can be assembled and solved in the form presented in Fig. 5. All unique nodes are concatenated into a single vector, and the local element matrices contribute to a global finite-element matrix, K. The linear system is solved twice: once for all the x values, and a sescond time for all the y values. The rectangular matrix P in Fig. 5 is a permutation matrix that has one entry per row with a value of 1 in a column corresponding to a constrained node. Therefore, P is a linear transformation that selects from the global vector of nodes only those that are on the boundary or are in the interior, but are fixed. In Fig. 5, the $*$ components of the solution vectors are essentially Lagrange multipliers so they are ignored, and the $\bar{x}$ and $\bar{y}$ sub-vectors represent the coordinates of the constrained nodes.

Figure 6 quantifies the change in mesh quality due to smoothing using one of the test cases shown earlier with varying mesh size. Four mesh quality metrics are used. For the determinant ratio metric, the determinant of the Jacobian of each element is computed at all the Gauss points of the element. The minimum determinant value is divided by the maximum within the element, and the smallest of these determinant ratios across the elements is reported, where 1 is the best possible value and a lower value indicates a mesh of lower quality. The minimum angle metric simply reports the minimum across all $4n$ angles, where n is the number of elements in the mesh, and a minimum angle close to 90° is desired. The aspect ratio is the largest side length divided by the smallest side length for an element, and a lower maximum aspect ratio in the mesh is preferable. The same is the case with the taper ratio, which is defined as the maximum side length plus the average side length, all divided by the average side length.

While there is no single metric that captures all the mesh quality features, the overall results in Fig. 6 suggest that the elliptical smoothing algorithm we propose is effective. The smoothing

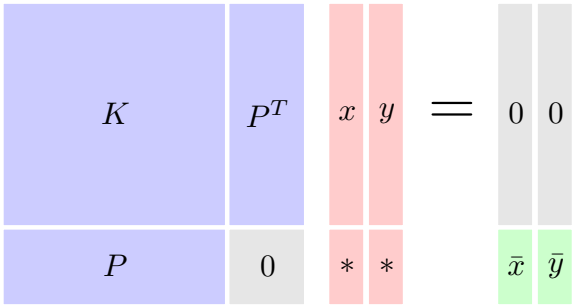


Figure 5: Global finite element equations for the elliptical smoothing. The K matrix represents the global finite element matrix, and P is a permutation matrix with a single entry of value 1 in each row, in the columns of nodes that are constrained. The vectors x, y represent the solution and $\bar{x}, \bar{y}$ contain the coordinates to which a node is constrained.

provides an improvement in nearly all the cases with respect to the minimum angle, aspect ratio, and taper ratio metrics. The determinant ratio metric shows a small decrease in quality, but this is a known side effect of elliptical smoothing [33].

4 Demonstration

In this section, we demonstrate the structural mesh generation algorithm on example problems. We show global aircraft meshes for several configurations for illustration, demonstrate the local mesh generation algorithm on example domains with different levels of resolution, and perform a mesh convergence study.

As stated in the introduction, the high-level motivation for this work is to facilitate the use of high-fidelity structural models when analyzing and comparing unconventional configurations. To this end, we show structural models for four configurations in Fig. 7. These meshes model ribs, spars, fuselage frames, and longerons, in addition to the skin.

Figure 8 focuses specifically on the local mesh generation algorithm, and shows the results for two example domains for three different resolution parameter values. For illustrative purposes, we show intermediate stages—the result after the CDT and the triangle merging step, which produces a quad-dominant mesh. The final mesh is the fully quad smoothed mesh. The internal edges in the two examples in Fig. 8 are chosen to create triangular and pentagonal regions to demonstrate the local mesh generation algorithm.

We also provide a mesh convergence study to verify that the unstructured quad meshing algorithm maintains element quality as the mesh is refined. The test case is the wing from the common research model (CRM) [34], which is a reference geometry developed for benchmarking. The CRM is based on the Boeing 777 wing, featuring a straight leading edge with the Yehudi break located at 37% of the span. The structural analyses are performed using the toolkit for the analysis of composite structures (TACS), which was developed by Kennedy and Martins [35].

The structure includes two main spars, stringers, ribs, and a secondary spar. The aft spar, the secondary spar, and one of the ribs form a triangle, testing the algorithm's ability to handle such cases. Even without the triangle, this case is designed to produce irregular nodes because of the wing taper—there will be far more elements in the chordwise direction at the root than at the tip of the wing.

A 1 kN point load is applied at the upper outboard-most tip of the front spar, and the vertical deflection is measured at the same location. An aluminum structure is used with a constant thick-

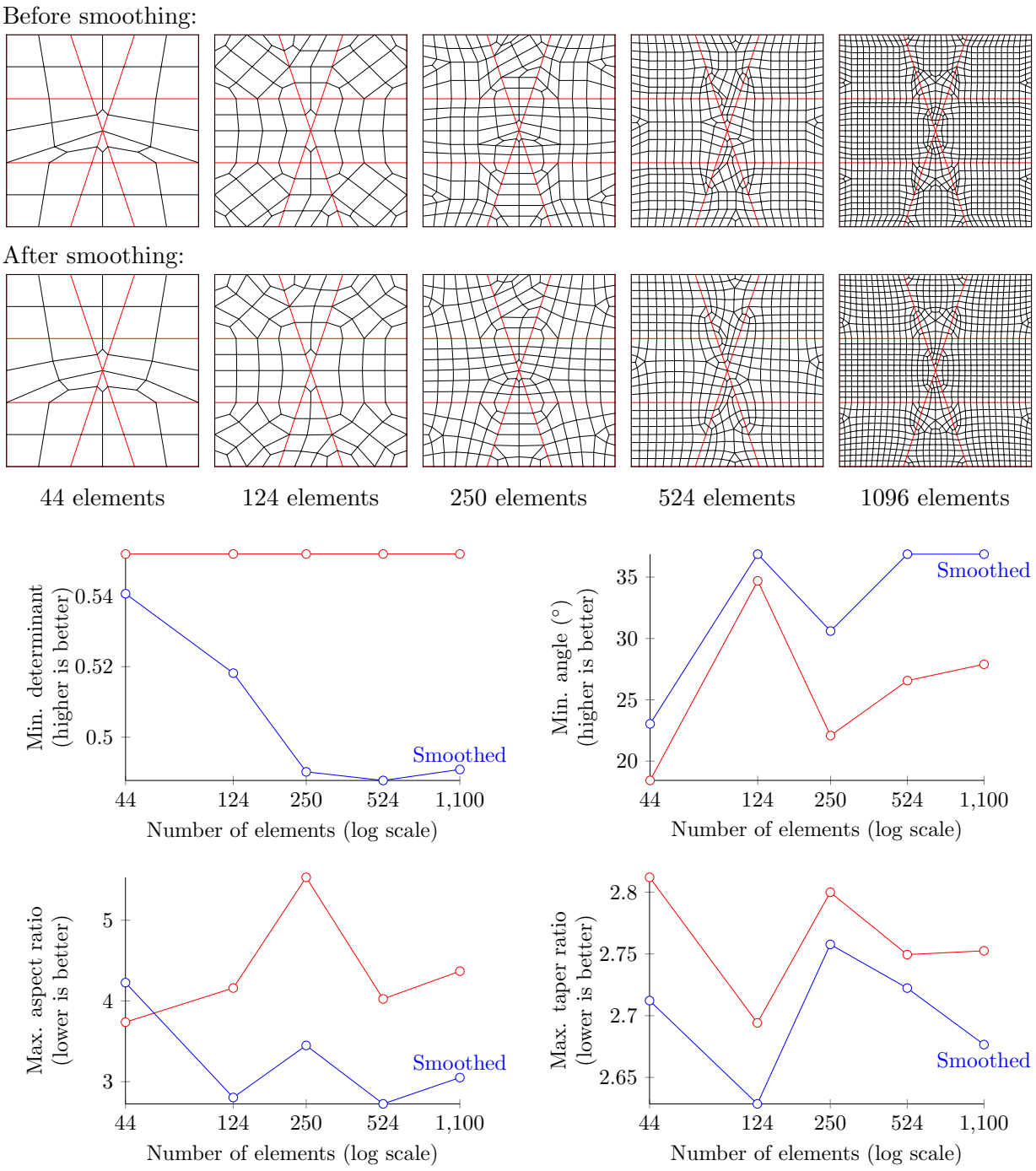
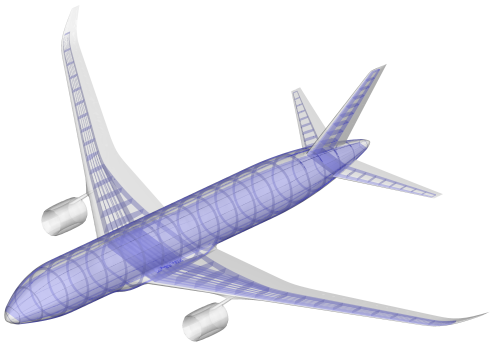
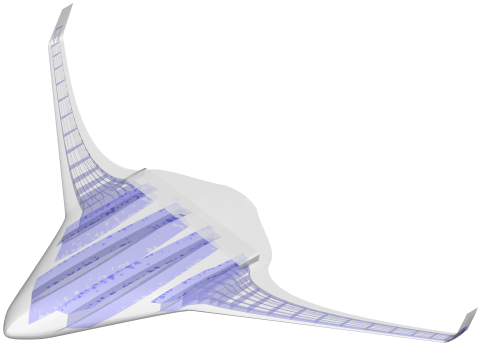


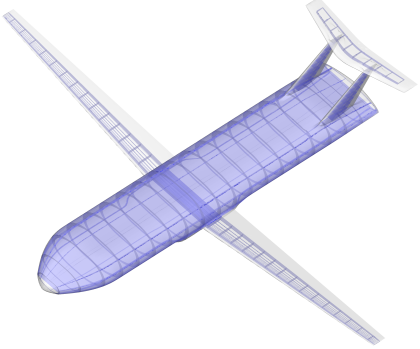
Figure 6: Comparison between the smoothed and unsmoothed meshes. In nearly all cases, the smoothed mesh shows improved quality for all but the Jacobian determinant metric, which is consistent with what is expected with Laplacian smoothing.



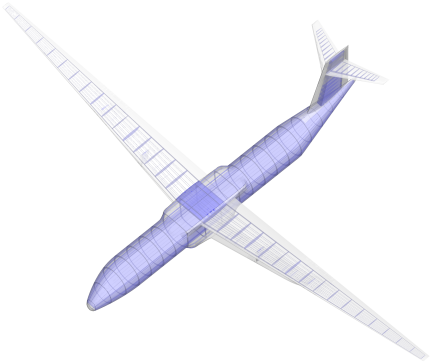
Conventional



Blended wing body



Double bubble



Truss-braced wing

Figure 7: Four configurations with the corresponding structural meshes.

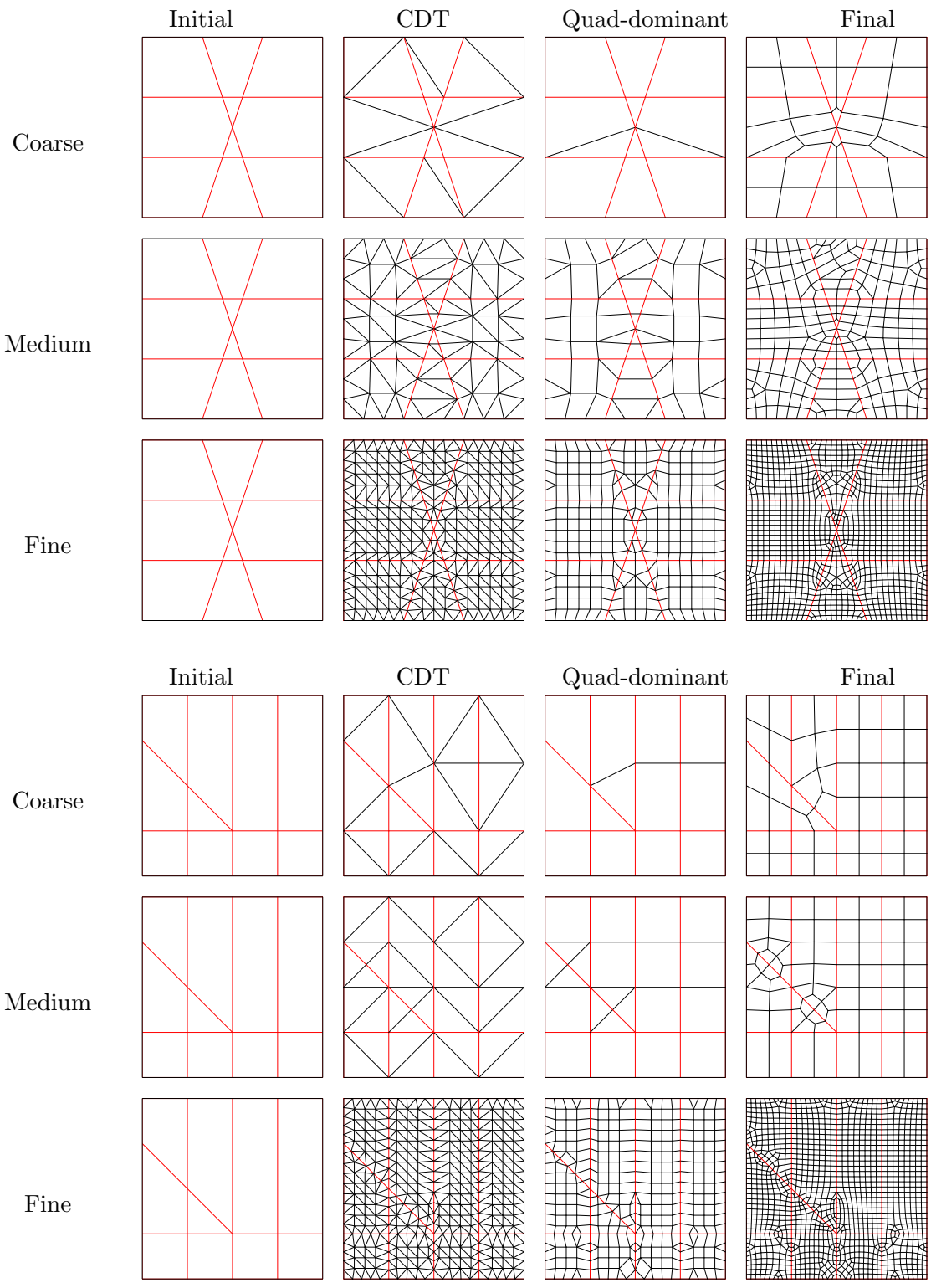


Figure 8: Demonstration of the local mesh generation algorithm on two example domains with three resolution parameters.

ness of 5 mm, and linear kinematics are assumed. Figure 9 shows six meshes of varying resolutions and the contours of vertical deflection, and Fig. 10 plots the results of the mesh convergence study. In Fig. 10, the displacement appears to converge monotonically to about 21 mm. This result suggests that the unstructured mesh generation algorithm maintains element quality as the mesh is refined.

5 Conclusion

The motivation for this work was to develop a tool for modeling aircraft structures to enable high-fidelity analysis and optimization. This tool can be applied during conceptual design when multiple configurations are being created and examined. We noted that such a tool does not currently exist because existing options fail to satisfy at least one of four requirements: meshing the full aircraft configuration simultaneously, being fully automated, morphing in response to geometry changes (e.g., for aerodynamic shape optimization), and having a differentiable mapping with efficient and accurate derivative computation.

In this paper, we presented an unstructured quadrilateral mesh generation algorithm for aircraft structures that satisfies all of these requirements. It defines the positions of internal structural nodes by interpolating points on the geometry. For the actual mesh generation, it divides the geometry into domains using the surfaces in the geometry representation, and applies a local mesh generation algorithm on each domain, independent from the others, using CDT and elliptical smoothing as key steps in the process. The entire algorithm is included in the GeoMACH aircraft geometry tool suite, which is available as open source software[2].

As implemented, the mesh generation algorithm benefits from the fact that GeoMACH provides a geometry composed of untrimmed quadrilateral surfaces. However, the overall algorithm can be extended to a more general geometry input. If, for instance, the geometry is a union of trimmed surfaces, the local mesh generation algorithm can be applied on each trimmed surface. For each domain, the trim curves would in general make the bounding edges curved even in parametric space. This would not pose a problem for the CDT, elliptical smoothing, or any of the other steps, and our algorithm could be extended to handle this. The only required addition would be a curve-curve intersection algorithm to determine if and where edges from imprinted features intersect with the trim curves.

Another promising extension of our algorithm would be to examine one of the many mesh clean-up algorithms that change the topology of the unstructured quad mesh to improve quality and increase the number of regular nodes. It would also be interesting to implement and consider alternative smoothing options for the purpose of improving mesh quality.

The proposed mesh generation tool makes it easier to develop relatively detailed structural models even for unconventional aircraft configurations. The expectation is that it will facilitate the use of higher-fidelity structural analysis tools earlier in the design process. Moreover, the automation offered by this tool could enable parametric studies on structural layout—for instance, one could set the number of ribs or the rib spacing in a wing as a design variable and carry out a study that performs structural or aerostructural optimization for each possible layout. Another specific study that this tool enables is aerostructural optimization of unconventional configurations with the entire airframe modeled structurally. This could help expedite the design process for new aircraft concepts and facilitate design studies that compare the performance of different unconventional configurations.

6 Conflict of interest statement

The authors declare that there is no conflict of interest in relation to this article.

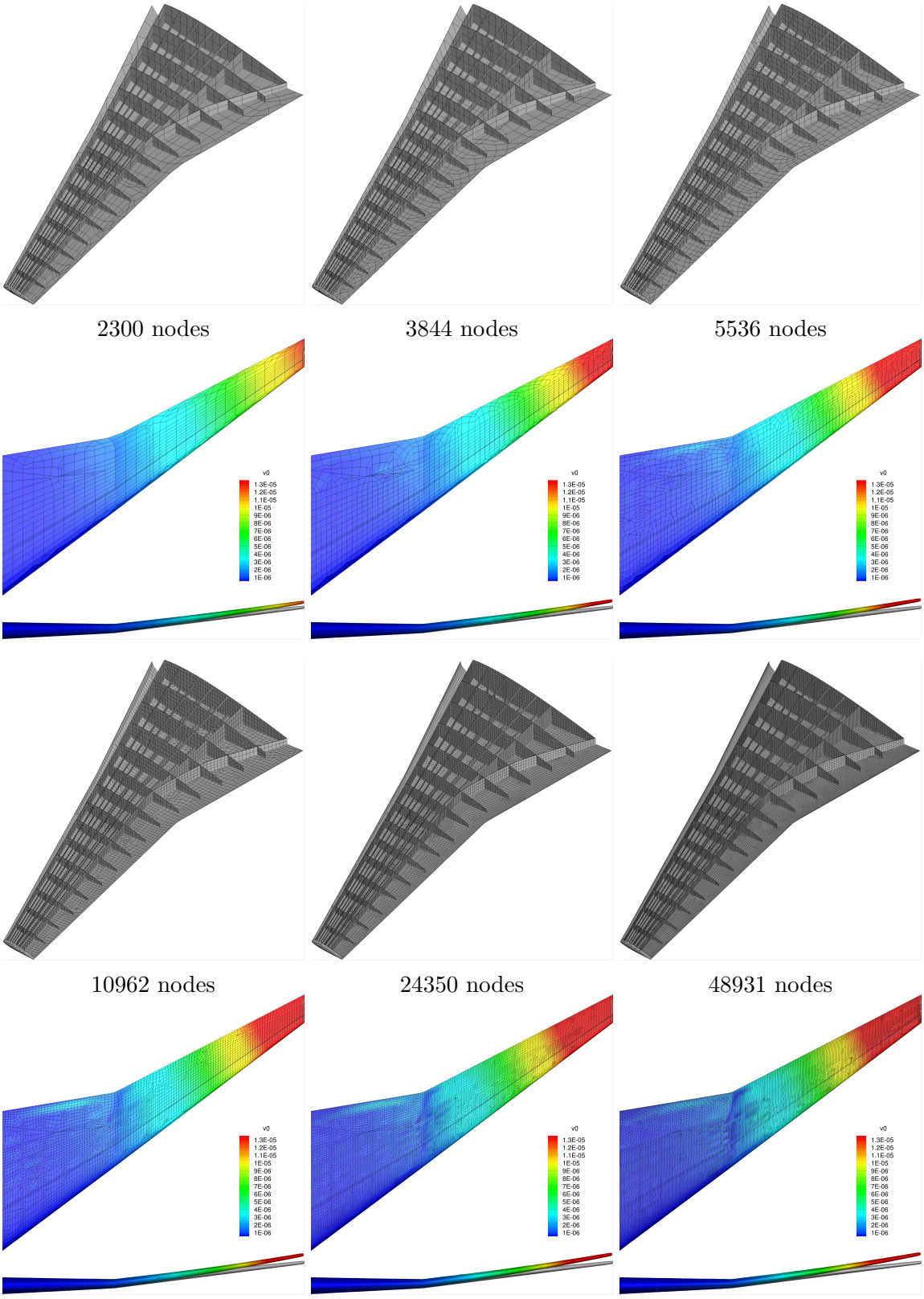


Figure 9: Meshes produced by the unstructured quad meshing algorithm and contours of vertical displacement.

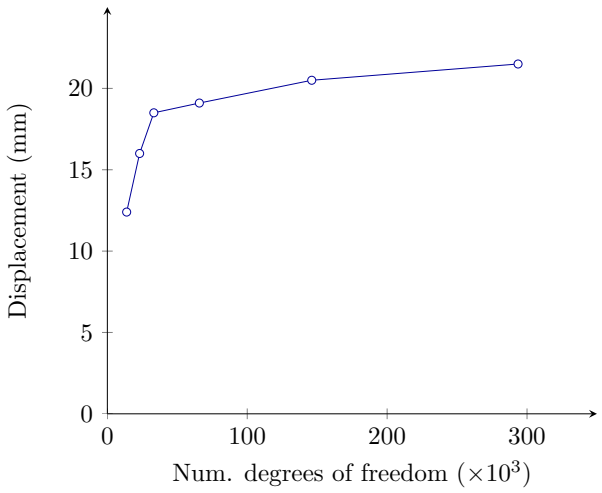


Figure 10: Mesh convergence study showing the vertical deflection of the upper, outboard-most node of the front spar for a 1 kN point load at the same node.

7 Acknowledgments

The authors gratefully acknowledge financial support from NASA through award No. NNX11AI19A.

8 Bibliography

References

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

[2] 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.

[3] Kenway, G. K. W. and Martins, J. R. R. A., "Multipoint 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.

[4] Kreisselmeier, G. and Steinhauser, R., "Systematic Control Design by Optimizing a Vector Performance Index," *International Federation of Active Controls Syposium on Computer-Aided Design of Control Systems, Zurich, Switzerland*, 1979.

[5] Poon, N. M. K. and Martins, J. R. R. A., "An Adaptive Approach to Constraint Aggregation Using Adjoint Sensitivity Analysis," *Structural and Multidisciplinary Optimization*, Vol. 34, No. 1, July 2007, pp. 61–73. doi:10.1007/s00158-006-0061-7.

[6] 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.

[7] Lyu, Z., Xu, Z., and Martins, J. R. R. A., "Benchmarking Optimization Algorithms for Wing Aerodynamic Design Optimization," *Proceedings of the 8th International Conference on Computational Fluid Dynamics*, Chengdu, Sichuan, China, July 2014, ICCFD8-2014-0203.

[8] Locatelli, D., Mulani, S. B., and Kapania, R. K., "Wing-Box Weight Optimization Using Curvilinear Spars and Ribs (SpaRibs)," *Journal of Aircraft*, Vol. 48, No. 5, 2011, pp. 1671–1684.

[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] 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.

[11] Kennedy, G. J. and Martins, J. R. R. A., "A Laminate Parametrization Technique for Discrete Ply Angle Problems with Manufacturing Constraints," *Structural and Multidisciplinary Optimization*, Vol. 48, No. 2, August 2013, pp. 379–393. doi:10.1007/s00158-013-0906-9.

[12] Brooks, T. R., Hwang, J. T., Kennedy, G. J., and Martins, J. R. R. A., "High-fidelity Structural optimization of a tow-steered composite wing," jun 2015.

[13] Brown, S. A., "Displacement Extrapolation for CFD+CSM Aeroelastic Analysis," *Proceedings of the 35th AIAA Aerospace Sciences Meeting*, Reno, NV, 1997, AIAA 1997-1090.

[14] Burdette, D. A., Kenway, G. K., and Martins, J., "Aerostructural design optimization of a continuous morphing trailing edge aircraft for improved mission performance," *17th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, American Institute of Aeronautics and Astronautics (AIAA), June 2016. doi:10.2514/6.2016-3209.

[15] 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.

[16] Hwang, J. T., Kenway, G. K. W., and Martins, J. R. R. A., "Geometry and Structural Modeling for High-Fidelity Aircraft Conceptual Design Optimization," *Proceedings of the 15th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, Atlanta, GA, June 2014.

[17] 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.

[18] 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.

[19] 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.

[20] 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.

[21] 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.

[22] 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.

[23] 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.

[24] 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.

[25] 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.

[26] 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.

[27] 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.

[28] Renka, R. J., "Algorithm 751: TRIPACK: a constrained two-dimensional Delaunay triangulation package," *ACM Transactions on Mathematical Software (TOMS)*, Vol. 22, No. 1, 1996, pp. 1–8.

[29] Parthasarathy, V. and Kodiyalam, S., "A constrained optimization approach to finite element mesh smoothing," *Finite Elements in Analysis and Design*, Vol. 9, No. 4, 1991, pp. 309–320.

[30] Thompson, J. F., Thames, F. C., and Mastin, C., "Automatic numerical generation of body-fitted curvilinear coordinate system for field containing any number of arbitrary two-dimensional bodies," *Journal of Computational Physics*, Vol. 15, No. 3, 1974, pp. 299–319. doi:http://dx.doi.org/10.1016/0021-9991(74)90114-4.

[31] Thompson, J. F., Soni, B. K., and Weatherill, N. P., *Handbook of grid generation*, CRC press, 1999.

[32] Winslow, A. M., "Numerical solution of the quasilinear Poisson equation in a nonuniform triangle mesh," *Journal of computational physics*, Vol. 1, No. 2, 1966, pp. 149–172.

[33] Finlayson, M. A., *ANSYS ICEM CFD User's Manual*, ANSYS, Inc., Canonsburg, PA.

[34] Vassberg, J. C., DeHaan, M. A., Rivers, S. M., and Wahls, R. A., "Development of a Common Research Model for Applied CFD Validation Studies," 2008, AIAA 2008-6919.

[35] 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.