A Novel Aerodynamic Shape Optimization Approach for Three-Dimensional Turbulent Flows

Lana Osusky* and David W. Zingg†

Institute for Aerospace Studies, University of Toronto, Toronto, Ontario, M3H 5T6, Canada

A novel gradient-based computational tool for efficient three-dimensional aerodynamic shape optimization is presented. An integrated approach is applied to geometry parameterization and mesh movement based on B-spline surfaces and volumes. Grid refinement and redistribution techniques are introduced to enable the fitting of B-spline volumes to grids appropriate for computing turbulent flows. The three-dimensional Reynolds-averaged Navier-Stokes equations are solved using a parallel Newton-Krylov-Schur flow analysis algorithm that makes use of Summation-by-Parts (SBP) operators and Simultaneous Approximation Terms (SATs) at block interfaces and boundaries. Gradients are evaluated using the discrete-adjoint method. The performance of the algorithm is demonstrated through a series of optimization studies, including a pitching moment coefficient optimization in subsonic flow and lift-constrained drag minimizations in transonic flow.

I. Introduction

In recent years, the airline industry has been facing financial pressure stemming from both increased attention to the importance of environmental stewardship in regulation and policy development and the rising cost of fuel. The latter has eclipsed labour costs to become the industry's dominant operational expense. The design of more efficient aerodynamic configurations can reduce the amount of fuel required for a given flight, reducing both greenhouse gas emissions and fuel costs. An efficient, high-fidelity numerical aerodynamic shape optimization algorithm would be a powerful tool for the designers of the next generation of aircraft, particularly as part of a multi-disciplinary optimization capability.

There are a number of gradient-free and gradient-based numerical optimization tools in use today, each of which comes with its own advantages and disadvantages. The choice of which type of method to implement is generally problem-dependent. Gradient-free methods, such as genetic algorithms,[1] are better at finding global optima, but are often time-consuming, particularly when the optimization problem has a large number of design variables. Gradient-based algorithms, such as the quasi-Newton method BFGS[2] for unconstrained optimization and SQP methods[3,4,5] for constrained optimization, are more efficient at finding local optima, although they may not find the global optimum. Chernukhin and Zingg[6] address this issue by developing two novel aerodynamic shape optimization strategies that make use of elements from genetic algorithms and gradient-based algorithms, along with a Sobol sequence sampling method to explore the design space for potential starting geometries for the optimizer.

Gradient-based optimization is not a new concept, but has evolved significantly to become a very popular approach to aerodynamic shape optimization, due in large part to the development of innovative gradient evaluation methods. Hicks *et al.*[7] calculated the gradient using finite-difference approximations, which placed limitations on the number of design variables that could be used. The introduction of adjoint-based methods by Pironneau[8] and Jameson[9] allows the gradient to be evaluated at a cost virtually independent of the number of design variables. In the discrete-adjoint approach, the adjoint equation is determined based on the discretized governing equations. In the continuous approach, the adjoint equation is derived before discretizing the governing equations and then discretized.

Jameson, Reuther, and colleagues applied the adjoint method to two-dimensional inviscid aerodynamic design problems[10] and then extended their work to three-dimensional constrained multi-point problems.[11,12,13]

*PhD Candidate, AIAA Student Member

†Professor and Director, Tier 1 Canada Research Chair in Computational Fluid Dynamics, J. Armand Bombardier Foundation Chair in Aerospace Flight, Associate Fellow AIAA

Jameson *et al.*[14] used the continuous adjoint approach to optimize wings and wing-body configurations based on the compressible Navier-Stokes equations. Anderson and Bonhaus[15] performed airfoil optimizations on unstructured grids using a discrete-adjoint approach to the Navier-Stokes equations coupled with the one-equation Spalart-Allmaras turbulence model. Nemec and Zingg[16,17] demonstrated the efficiency of a Newton-Krylov scheme in performing two-dimensional aerodynamic shape optimization based on the Navier-Stokes equations and the Spalart-Allmaras turbulence model. Laminar-turbulent transition prediction was incorporated into a gradient-based Newton-Krylov optimization algorithm by Driver and Zingg[18] and was used to design a series of high-lift airfoils.

We use an integrated geometry parameterization and mesh movement scheme which fits the blocks of the computational mesh with B-spline volumes.[19] The B-spline control points are moved based on the principles of linear elasticity, while the fine mesh is updated algebraically based on the B-spline volume basis functions. Flow analysis is carried out using a parallel Newton-Krylov algorithm with approximate-Schur preconditioning.[20] Gradients are evaluated using the discrete-adjoint approach. The gradient-based SQP optimization algorithm SNOPT[3] is used to update the design variables subject to linear or nonlinear constraints. The objective of the present work is to extend the Euler-based aerodynamic shape optimization algorithm of Hicken and Zingg[19,21] to handle turbulent flows governed by the Reynolds-Averaged Navier-Stokes equations and to characterize the resulting algorithm.

The main components of the algorithm are discussed in Sections II to V. More specifically, the geometry parameterization and mesh movement are described in Section II. Section III discusses some issues related to the fitting and movement of finely-spaced meshes suitable for modelling turbulent flow and how they can be overcome. Section IV reviews the Newton-Krylov flow solver used to solve the three-dimensional Reynolds-Averaged Navier-Stokes equations. The process used to compute the gradients is broken down in Section V. Sample optimization results are presented in Section VI, followed by our conclusions in Section VII.

II. Geometry Parameterization and Mesh Movement

Using the integrated parameterization and mesh movement approach developed by Hicken and Zingg,[19] an initial multi-block volume mesh is mapped with B-spline volumes. These mappings are defined by a tensor product of the control points, $\mathbf{B}_{ijk}$, and their p^{th}-order basis functions, $\mathcal{N}^{(p)}$, and are of the form

$$\mathbf{x}(\boldsymbol{\xi}) = \sum_{i=1}^{N_i}\sum_{j=1}^{N_j}\sum_{k=1}^{N_k}\mathbf{B}_{ijk}\mathcal{N}_i^{(p)}(\xi)\,\mathcal{N}_j^{(p)}(\eta)\,\mathcal{N}_k^{(p)}(\zeta)\,, \tag{1}$$

where $\mathbf{x}(\boldsymbol{\xi})$ represents the set of Cartesian coordinates of the B-spline volume as a function of the set of curvilinear coordinates $\boldsymbol{\xi} = (\xi,\eta,\zeta) \in \mathbb{R}^3 | \xi,\eta,\zeta \in [0,1]$. The basis functions in the ξ-direction are expressed as

$$\mathcal{N}_i^{(1)}(\xi;\eta,\zeta) = \begin{cases} 1 & \text{if } T_i(\eta,\zeta) \le \xi < T_{i+1}(\eta,\zeta)\,, \\ 0 & \text{otherwise} \end{cases}$$

$$\begin{aligned}\mathcal{N}_i^{(p)}(\xi;\eta,\zeta) = &\left(\frac{\xi - T_i(\eta,\zeta)}{T_{i+p-1}(\eta,\zeta) - T_i(\eta,\zeta)}\right)\mathcal{N}_i^{(p-1)}(\xi;\eta,\zeta) \\ &+ \left(\frac{T_{i+p}(\eta,\zeta) - \xi}{T_{i+p}(\eta,\zeta) - T_{i+1}(\eta,\zeta)}\right)\mathcal{N}_{i+1}^{(p-1)}(\xi;\eta,\zeta).\end{aligned} \tag{2}$$

Similar expressions exist for the basis functions in the η- and ζ-directions, $\mathcal{N}_j^{(p)}(\eta;\zeta,\xi)$ and $\mathcal{N}_k^{(p)}(\zeta;\xi,\eta)$, respectively. The knot values, $T_i(\eta,\zeta)$, in the interior of the B-spline volume are given by

$$T_i(\eta,\zeta) = [(1-\eta)(1-\zeta)]T_{i,(0,0)} + [\eta(1-\zeta)]T_{i,(1,0)} + [(1-\eta)\zeta]T_{i,(0,1)} + [\eta\zeta]T_{i,(1,1)}. \tag{3}$$

Again, similar expressions denote the knot values in the η- and ζ-directions, $T_i(\zeta,\xi)$ and $T_i(\xi,\eta)$. The first p and last p knot values for a given B-spline curve are set to 0 and 1, respectively; the edge knot values, $T_{i,(0,0)}$, $T_{i,(1,0)}$, $T_{i,(0,1)}$, and $T_{i,(1,1)}$ are constants. Given the edge knot values, we can bilinearly interpolate the interior knots from (3). We then use a least-squares fitting routine to determine the locations

of the B-spline control points. This is done in a sequential manner, first fitting the block edges, then the sides, and finally the internal volumes. Since the parameter values ξ, η, and ζ are based on a chord-length parameterization, the knot values are inherently chord-length-based as well. It is this property which results in a coarse B-spline volume mesh that mimics the spacing of the original computational mesh.

In an aerodynamic optimization, the control points corresponding to the surface of the geometry are used as design variables. When the coordinates of the surface control points are changed by the optimizer, the remaining internal control points in the B-spline volume are updated based on the principles of linear elasticity:[19,22]

$$\boldsymbol{\mathcal{M}}^{(i)}(\mathbf{b}^{(i-1)}, \mathbf{b}^{(i)}) = K^{(i)}[\mathbf{b}^{(i)} - \mathbf{b}^{(i-1)}] - \mathbf{f}^{(i)} = 0, \qquad i = 1, ..., m \tag{4}$$

where $\boldsymbol{\mathcal{M}}^{(i)}$ is the mesh movement residual, $\mathbf{b}^{(i)}$ the set of B-spline control point coordinates, K the global stiffness matrix at increment i, and m the number of increments. We typically perform the control mesh updates of (4) in five increments, which strikes a balance between speed and robustness in accommodating large changes in the geometry.[19] Note that $\mathbf{B}_{ijk}$ in (1) represents a set of coordinates for a single control point, whereas $\mathbf{b}^{(i)}$ is a block-column vector of the coordinates for all control points in the B-spline volume. The force vector $\mathbf{f}^{(i)}$ is defined implicitly by the surface control point coordinates. We solve (4) using the conjugate gradient method preconditioned with ILU(p), where the fill level, p, is 1.

While the linear-elasticity approach to mesh movement is a robust one, applying it to the actual computational mesh is time-consuming. We instead apply the linear-elasticity method to the coarse mesh made up of the B-spline control points, which is typically 2-3 orders of magnitude smaller than the fine computational mesh. The fine mesh is updated using an algebraic approach based on the B-spline volume basis functions. This integrated approach allows us to perform accurate and efficient mesh updates that maintain good mesh quality and require negligible CPU time compared to the time required to obtain a converged flow solution. It also greatly reduces the cost of solving the mesh adjoint system (see Section V).

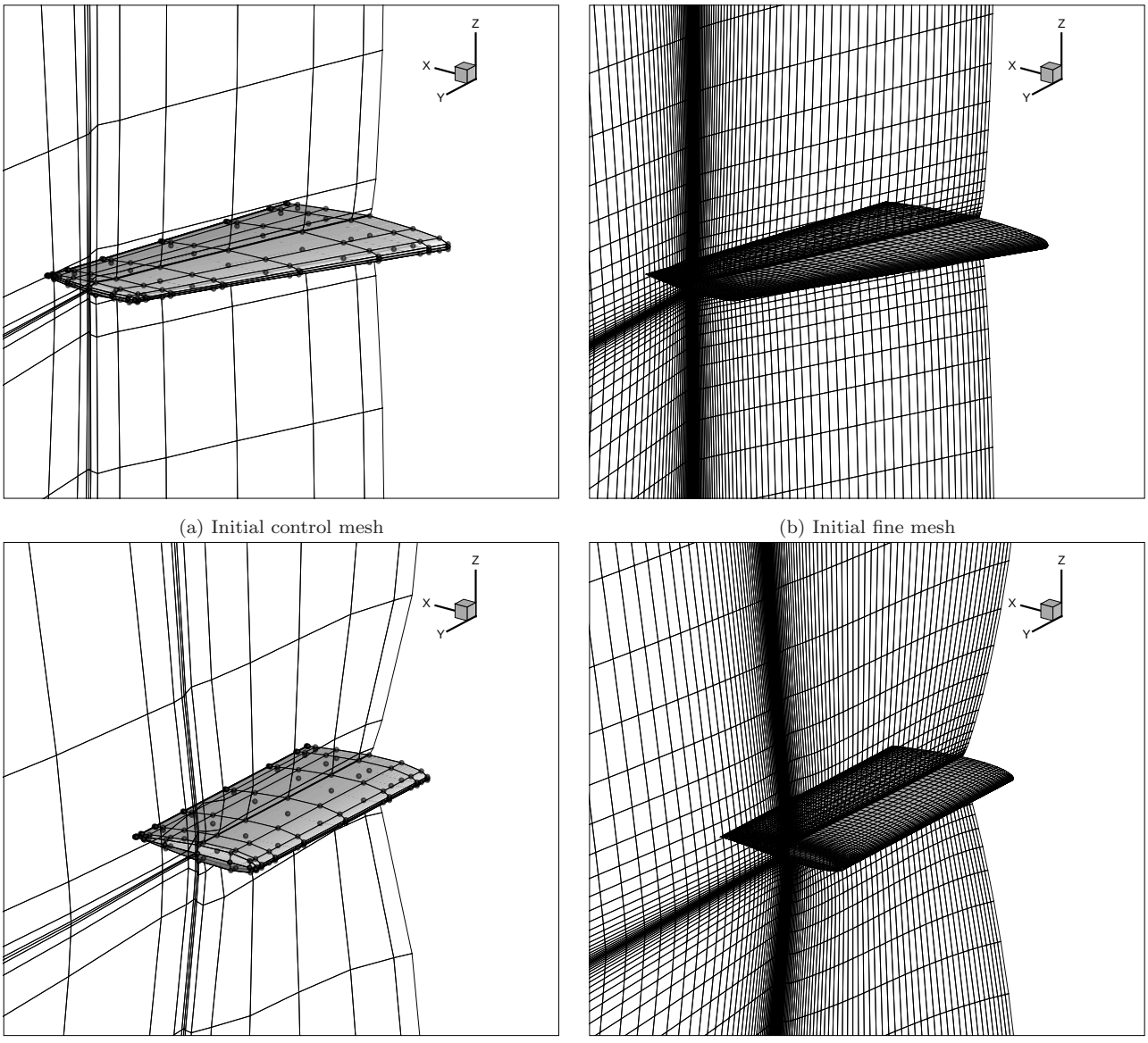
The mesh movement algorithm is illustrated in Figure 1, where the control point and fine computational meshes are shown for the case of an ONERA M6 wing transforming into a rectangular wing with NACA0012 sections. Each of the 12 blocks of the initial mesh contain $45 \times 65 \times 33$ nodes, while the corresponding control mesh volumes are made up of $9 \times 9 \times 9$ nodes each.

III. Fitting B-Spline Volume Meshes

The main differences between a mesh used for an inviscid optimization and one used for a turbulent case are in the node spacings in various regions of the grid, particularly in the off-wall direction. Inviscid meshes typically have an off-wall spacing of approximately 10^{-3} chord units; this spacing would be inappropriate for a turbulent flow analysis, as it would not resolve the flow features close to the aerodynamic surfaces. A more appropriate off-wall spacing for a turbulent analysis is approximately 10^{-6} chord units. Unfortunately, the RMS error for a mesh obtained from the B-spline volume fitting is of the order of 10^{-3} chord units. When we attempt to fit meshes with spacings that are of a lower magnitude than the fitting error, the B-spline control points bunch together and overlap.

Figure 2a shows the control point distribution and fitted mesh at the symmetry plane of the inviscid mesh, focusing on the leading edge. The mesh used in this example is a 12-block ONERA M6 mesh with 2.47 million nodes and an off-wall distance of 10^{-3} chord units. The B-spline volumes are fit with $9 \times 9 \times 5$ control points. The red control points correspond to the block containing the wing surface, while the black control points correspond to the neighbouring block. Figures 2b and 2c show two views of the deteriorating quality of the B-spline volume mesh near the leading edge when we attempt to fit a 12-block ONERA M6 mesh with same number of nodes as the inviscid grid, but with an off-wall spacing of 10^{-4} chord units, which is only one order of magnitude finer than that of the inviscid mesh. Even though the off-wall, leading edge, and wing tip spacing are only up to an order of magnitude finer compared to the inviscid mesh, we see that the control points from the two blocks overlap near the leading edge. The problem only intensifies as the off-wall distance decreases. The ideal solution would be to obtain a B-spline fit for an inviscid mesh and subsequently achieve a refined computational mesh spacing appropriate for turbulent analysis.

To this end, we introduce a dual-option mesh refinement process that can be applied to coarse meshes; the B-spline control point distribution is obtained from the original coarse mesh and is unaffected by the subsequent refinement strategy. At the beginning of any optimization, each block of the computational mesh is fit with B-spline volumes using the least-squares fitting routine described in Section II. After the



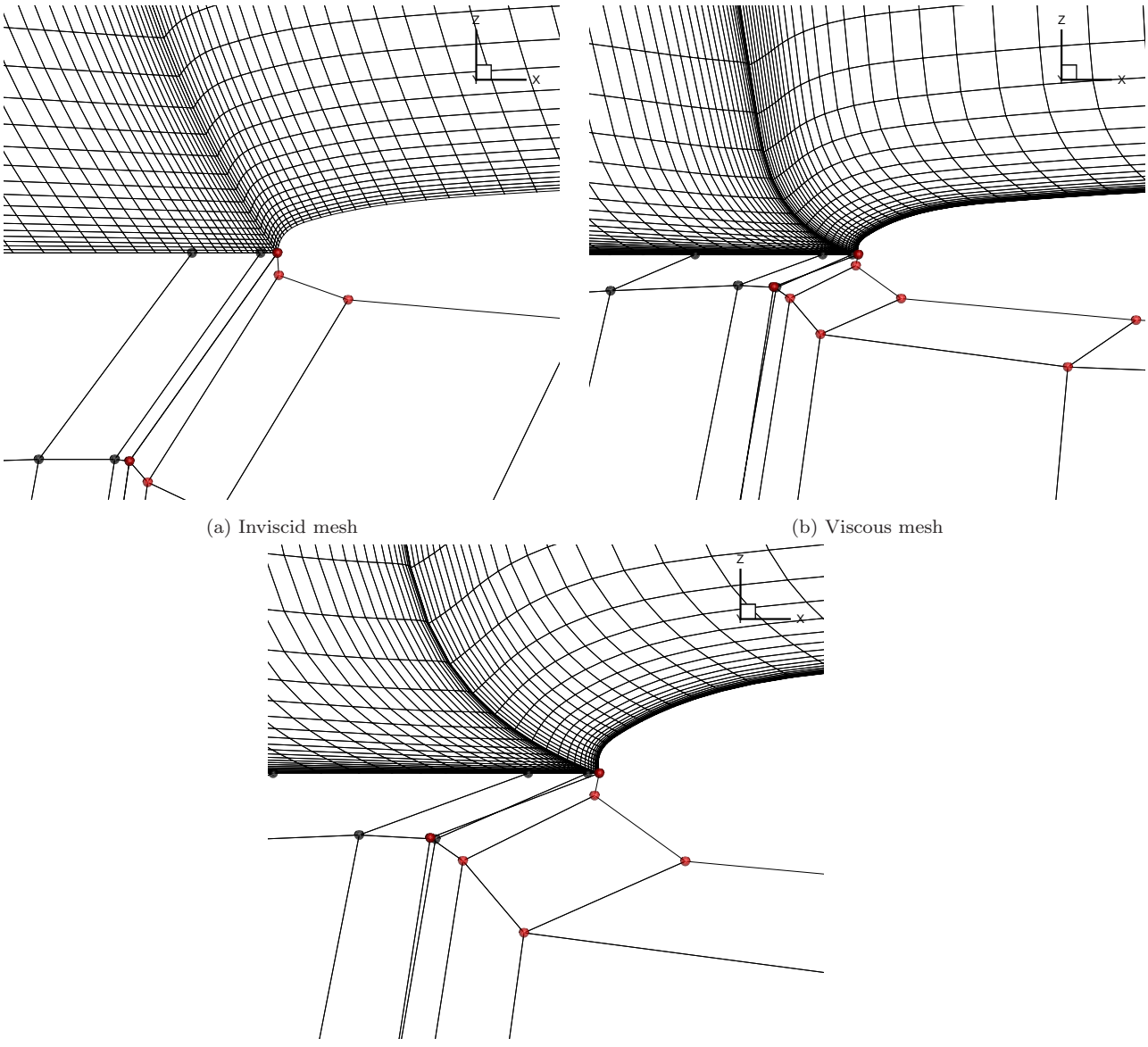
(a) Initial control mesh

(b) Initial fine mesh

(c) Final control mesh

(d) Final fine mesh

Figure 1: Example of B-spline mesh movement



(a) Inviscid mesh

(b) Viscous mesh

(c) Detail of viscous mesh

Figure 2: Symmetry plane control point distribution for inviscid and viscous meshes

Table 1: 12-block ONERA M6 mesh spacing data

Refine option	**Grid size**	**Spacing**		
	(Nodes)	Off-wall	LE and TE	Spanwise tip
None	2 470 500	1.97×10^{-3}	1.98×10^{-3}	9.70×10^{-4}
Resize	4 132 092	1.80×10^{-3}	1.75×10^{-3}	8.42×10^{-4}
Refine	2 470 500	4.88×10^{-7}	2.34×10^{-5}	1.08×10^{-5}
Combination	4 132 092	3.99×10^{-7}	1.90×10^{-5}	8.89×10^{-6}

B-spline volume mesh has been generated, we can choose to refine the mesh using a node-insertion, or grid refinement, method. The grid refinement option increases the number of nodes in each coordinate direction by a scaling factor (1.2 is used for the examples presented in this paper). The entire set of parameter values is re-evaluated using the chord-length based parameterization so that the existing nodes are redistributed to account for the additional nodes.

The second refinement option is grid redistribution, and involves the targeted refinement of the spacing-control function parameters along specific grid edges. Starting with any fitted computational mesh (which may or may not have already made use of the grid resizing option), the edge spacing parameters corresponding to specific blocks can be refined by a set of user-defined scaling factors. The parameter values $\boldsymbol{\xi} = (\xi, \eta, \zeta)$ throughout the rest of the grid are re-evaluated based on the edge parameters so that we need only refine the block edges in order to achieve a distributed refinement. The important point to note about using either or both of these refinement options is that they are done after the B-spline volume mesh has been calculated, so that any refinement that is performed to obtain finer mesh spacings from a coarse mesh has no effect on the control mesh.

Examples are presented to show the effectiveness of the dual-option refinement scheme. All examples start with the coarse 12-block ONERA M6 grid detailed in Table 1. The mesh spacing obtained using the various options of the refinement scheme are shown in Figure 3, where we focus on the spacing at the root chord. The same number of streamwise B-spline control points is is used for each example. Figure 3a shows the initial spacing at the root chord of the coarse grid, with the leading edge detail shown in the inset image. Figures 3b and 3c shows the spacing resulting from the grid refinement and grid redistribution options, respectively, and Figure 3d shows the spacing obtained from applying both the grid refinement and grid redistribution options. This strategy provides a means of obtaining a well-spaced B-spline volume mesh while providing the refined node spacing required for analyzing turbulent flows for the purpose of optimization. The spacings of the coarse mesh and those of the refined meshes are shown in Table 1.

IV. Newton-Krylov-Schur Flow Solver

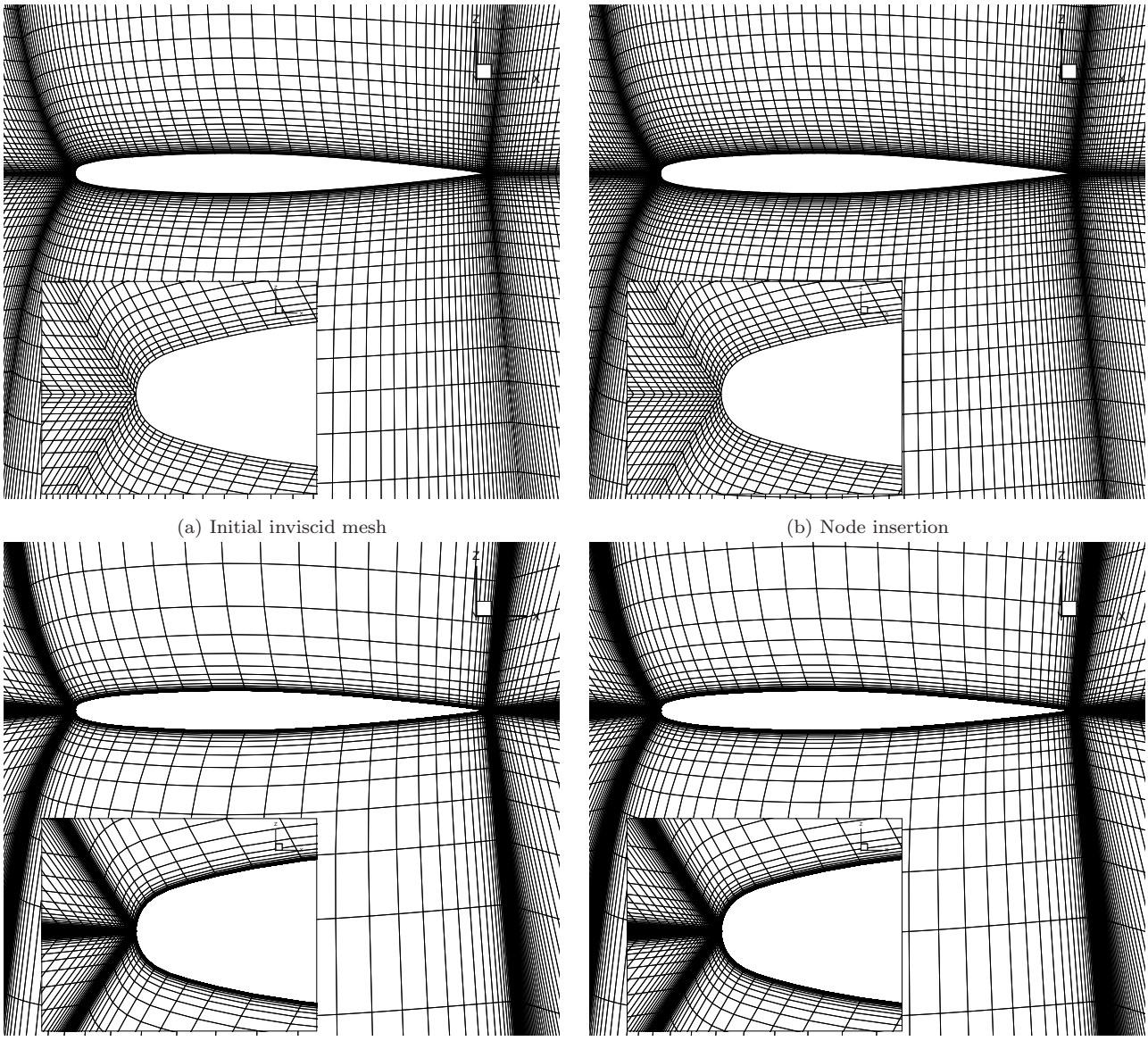
We solve the three-dimensional Reynolds-averaged Navier-Stokes equations, which, when transformed into curvilinear coordinates, are given by

$$\partial_t \hat{\mathbf{Q}} + \partial_\xi \hat{\mathbf{E}} + \partial_\eta \hat{\mathbf{F}} + \partial_\zeta \hat{\mathbf{G}} = \frac{1}{Re}\left(\partial_\xi \hat{\mathbf{E}}_\mathbf{v} + \partial_\eta \hat{\mathbf{F}}_\mathbf{v} + \partial_\zeta \hat{\mathbf{G}}_\mathbf{v}\right), \tag{5}$$

where $\hat{\mathbf{Q}}$ represents the conservative flow variables, and Re is the Reynolds number. The inviscid fluxes are given by $\hat{\mathbf{E}}$, $\hat{\mathbf{F}}$, and $\hat{\mathbf{G}}$, and the viscous fluxes by $\hat{\mathbf{E}}_\mathbf{v}$, $\hat{\mathbf{F}}_\mathbf{v}$, and $\hat{\mathbf{G}}_\mathbf{v}$. The flow solver is based on the Euler solver developed by Hicken and Zingg,[20] which was extended to include viscous and turbulent effects by Osusky *et al.*[23] Turbulence is modelled using the Spalart-Allmaras one-equation turbulence model,[24] given by

$$\frac{\partial \tilde{\nu}}{\partial t} = M(\tilde{\nu})\tilde{\nu} + P(\tilde{\nu})\tilde{\nu} - D(\tilde{\nu})\tilde{\nu} + T, \tag{6}$$

where $M(\tilde{\nu})\tilde{\nu}$ represents advection and diffusion, $P(\tilde{\nu})\tilde{\nu}$ is a production source term, $D(\tilde{\nu})\tilde{\nu}$ is a wall de-



(a) Initial inviscid mesh

(b) Node insertion

(c) Spacing refinement

(d) Combination

Figure 3: Root chord meshes for inviscid and refined grids

struction source term, and T is a trip function representing the transition point at which the flow becomes turbulent. The trip term is not used in the examples shown here; fully turbulent flow is assumed.

The governing equations are discretized using second-order-accurate summation-by-parts (SBP) operators[25] with scalar artificial dissipation. Boundary conditions are enforced using Simultaneous Approximation Terms (SATs).[26] Additionally, SATs are used to enforce the coupling of adjoining blocks of the computational mesh,[27,28,29] and are an attractive option in minimizing inter-processor communication, as solving the governing equations on any given block requires less information from neighbouring blocks compared to the more common approach using halo nodes. Moreover, with the SAT treatment, slope continuity of the mesh is not needed at the mesh block interfaces, which simplifies mesh generation.

Discretizing the governing equations produces a system of nonlinear equations

$$\boldsymbol{\mathcal{R}}(\mathbf{v}, \mathbf{b}^{(m)}, \mathbf{q}) = 0, \tag{7}$$

where $\mathbf{v}$ is the set of design variables, which can include a subset of the surface B-spline control points, the angle of attack, and planform design variables (e.g. sweep), $\mathbf{b}^{(m)}$ contains the control point coordinates of the B-spline volumes after the final increment of the mesh movement, and $\mathbf{q}$ is a block-column vector of conservative flow variables.

We solve (7) using a parallel Newton-Krylov algorithm. At each Newton iteration, n, we solve the sparse linear system given by

$$A^{(n)} \Delta \mathbf{q}^{(n)} = -\boldsymbol{\mathcal{R}}^{(n)}, \tag{8}$$

where $\boldsymbol{\mathcal{R}}^{(n)} = \boldsymbol{\mathcal{R}}(\mathbf{v}, \mathbf{b}^{(m)}, \mathbf{q}^{(n)})$, $\Delta \mathbf{q}^{(n)} = \mathbf{q}^{(n+1)} - \mathbf{q}^{(n)}$, and the Jacobian matrix $A^{(n)}$ is

$$A_{ij}^{(n)} = \frac{\partial \mathcal{R}_i^{(n)}}{\partial q_j}. \tag{9}$$

We solve (8) using flexible GMRES with approximate-Schur preconditioning. An approximate-Newton start-up phase is used to determine a suitable initial iterate. Once the residual has been reduced by three orders of magnitude, the Newton-Krylov algorithm enters the inexact-Newton phase, which converges the residual norm to a relative tolerance of 10^{-10}. Deep convergence of the flow solution aids in the convergence of the optimizer.

V. Gradient Evaluation

We use the gradient-based sequential quadratic programming optimization algorithm SNOPT.[3] As with any gradient-based optimization method, an accurate and efficient means of calculating the gradient is required in order for the method to be effective. Finite differencing[7] is not a viable option for large-scale problems with many design variables. The adjoint method used by Pironneau[30] and Jameson et al.[14] is a better option, as the time required to complete one computation of the gradient is virtually independent of the number of design variables. This is a major advantage for optimization problems with a large number of design variables. The discrete approach to the adjoint method, developed based on the discretized form of the governing equations, is used.

We wish to minimize an objective function, $\mathcal{J}$, such as drag, which is a function of the design variables, denoted by $\mathbf{v}$, as well as the grid and flow properties. The problem is subject to the mesh and flow residual equations, which we treat as constraints that are solved outside of SNOPT by minimizing the Lagrangian function:

$$\mathcal{L}(\mathbf{v}, \mathbf{b}^{(m)}, \mathbf{q}, \boldsymbol{\lambda}^{(i)}|_{i=1}^{m}, \boldsymbol{\psi}) = \mathcal{J}(\mathbf{v}, \mathbf{b}^{(m)}, \mathbf{q}) + \sum_{i=1}^{m} \boldsymbol{\lambda}^{(i)T} \boldsymbol{\mathcal{M}}^{(i)}(\mathbf{v}, \mathbf{b}^{(i-1)}, \mathbf{b}^{(i)}) + \boldsymbol{\psi}^T \boldsymbol{\mathcal{R}}(\mathbf{v}, \mathbf{b}^{(m)}, \mathbf{q}), \tag{10}$$

where the Lagrange multipliers $\boldsymbol{\lambda}^{(i)}|_{i=1}^{m}$ and $\boldsymbol{\psi}$ represent adjoint variables for the mesh movement and flow residual equations, respectively.

The gradient evaluation is performed in a sequential manner. After solving (7) for the flow variables, $\mathbf{q}$, we solve the flow adjoint system, given by

$$\left(\frac{\partial \boldsymbol{\mathcal{R}}}{\partial \boldsymbol{q}}\right)^T \boldsymbol{\psi} = -\left(\frac{\partial \mathcal{J}}{\partial \boldsymbol{q}}\right)^T, \tag{11}$$

for the vector of flow adjoint variables, $\boldsymbol{\psi}$. In the Jacobian matrix of the flow residual $\frac{\partial \boldsymbol{\mathcal{R}}}{\partial \boldsymbol{q}}$, the discrete Euler and viscous fluxes, as well as the numerical dissipation, are linearized analytically, while the inviscid SATs that enforce boundary conditions and block interface continuity are differentiated using the complex-step method.[31] The viscous and turbulent SATs are differentiated using both the complex-step and analytical approaches.[23] The flow adjoint system is solved using a simplified and flexible variant of GCROT (Generalized Conjugate Residual with Orthogonalization and Truncation), a nested GMRES-type solver that recycles Krylov subspaces in order to reduce memory requirements.[32]

We next obtain the mesh adjoint variables corresponding to the final increment of mesh movement, $\boldsymbol{\lambda}^{(m)}$, from

$$\left(\frac{\partial \boldsymbol{\mathcal{M}}^{(m)}}{\partial \boldsymbol{b}^{(m)}}\right)^T \boldsymbol{\lambda}^{(m)} = -\left(\frac{\partial \mathcal{J}}{\partial \boldsymbol{b}^{(m)}}\right)^T - \left(\frac{\partial \boldsymbol{\mathcal{R}}}{\partial \boldsymbol{b}^{(m)}}\right)^T \boldsymbol{\psi}. \tag{12}$$

We expand the right-hand side of (12) using the chain rule, which gives

$$-\left(\frac{\partial \mathcal{J}}{\partial \boldsymbol{b}^{(m)}}\right)^T - \left(\frac{\partial \boldsymbol{\mathcal{R}}}{\partial \boldsymbol{b}^{(m)}}\right)^T \boldsymbol{\psi} = -\left(\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{b}^{(m)}}\right)^T \left[\left.\frac{\partial \mathcal{J}}{\partial \boldsymbol{g}}\right|_{\boldsymbol{m}} + \left(\left.\frac{\partial \mathcal{J}}{\partial \boldsymbol{m}}\right|_{\boldsymbol{g}} + \boldsymbol{\psi}^T \frac{\partial \boldsymbol{\mathcal{R}}}{\partial \boldsymbol{m}}\right) \frac{\partial \boldsymbol{m}}{\partial \boldsymbol{g}}\right]^T. \tag{13}$$

This reformulation gives us a system that is not difficult to solve and only requires the storage of vector-matrix and matrix-vector products. The vectors $\boldsymbol{g}$ and $\boldsymbol{b}$ represent the Cartesian grid coordinates and B-spline volume control point coordinates, respectively. The $\left.\frac{\partial \mathcal{J}}{\partial \boldsymbol{g}}\right|_{\boldsymbol{m}}$ term represents the partial derivative of the objective function with respect to the grid points with the metric terms arising from the curvilinear coordinate transformation frozen, while the $\left.\frac{\partial \mathcal{J}}{\partial \boldsymbol{m}}\right|_{\boldsymbol{g}}$ term represents the partial derivative of the objective function with respect to the metrics with the grid coordinates frozen. The partial derivatives on the right-hand side of (13) are obtained analytically, and the left-hand side can be expressed as the symmetric stiffness matrix at increment m, $\boldsymbol{K}^{(m)}$. The system is solved using the preconditioned conjugate gradient method.

Once the vector of mesh adjoint variables at the last mesh movement increment has been computed, the mesh adjoint variables corresponding to the remaining increments, $\{\boldsymbol{\lambda}^{(i)}\}_{i=1}^{(m-1)}$, can be obtained from

$$\left(\frac{\partial \boldsymbol{\mathcal{M}}^{(i)}}{\partial \boldsymbol{b}^{(i)}}\right)^T \boldsymbol{\lambda}^{(i)} = -\left(\frac{\partial \boldsymbol{\mathcal{M}}^{(i+1)}}{\partial \boldsymbol{b}^{(i)}}\right)^T \boldsymbol{\lambda}^{(i+1)}, i \in \{m-1, m-2, \ldots, 1\}. \tag{14}$$

The left-hand side of (14) can be expressed as the symmetric stiffness matrix at increment i, $\boldsymbol{K}^{(i)}$. The right-hand side matrix is obtained using the complex-step method, which requires little computational time, as it is only applied to the coarse control mesh. The system is solved using the preconditioned conjugate gradient method.

Following the computation of the adjoint variables from systems (11) - (14), the gradient of the Lagrangian function with respect to the design variables is computed from

$$\mathcal{G} = \frac{\partial \mathcal{J}}{\partial \mathbf{v}} + \sum_{i=1}^{m} \left(\boldsymbol{\lambda}^{(i)T} \frac{\partial \boldsymbol{\mathcal{M}}^{(i)}}{\partial \mathbf{v}}\right) + \boldsymbol{\psi}^T \frac{\partial \boldsymbol{\mathcal{R}}}{\partial \mathbf{v}}. \tag{15}$$

The sequential process of computing the gradient must be repeated for any non-linear constraints, such as lift. This process is not necessary for linear constraints, as they are satisfied exactly by SNOPT.

VI. Results

A. Inverse Design

To test the optimization algorithm, we perform a simple inverse optimization with the ONERA M6 wing. The 3 coordinates of a control point on the upper surface of the wing serve as design variables, along with the angle of attack, which is initially set to 2 degrees. A 12-block grid made up of 2.47 million nodes is used; each block of the computational mesh is parameterized by $9 \times 9 \times 5$ B-spline control points. The off-wall mesh spacing is 2.35×10^{-6} root chord units. The flow is fully turbulent, with a Mach number of 0.5 and a Reynolds number of 3 million.

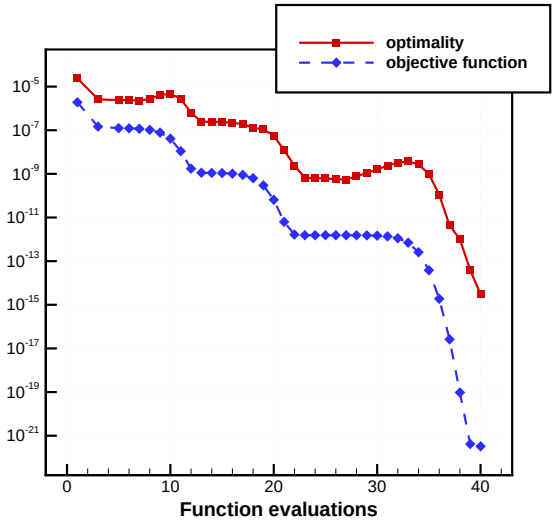


Figure 4: Convergence history for inverse design case

The four design variables are randomly perturbed, and the target surface pressure distribution is obtained from the perturbed geometry. The objective function to be minimized is given by

$$\mathcal{J} = \frac{1}{2} \sum_{i=1}^{N_{surf}} \left(p_i - p_{i,targ}\right)^2 \Delta A_i, \tag{16}$$

where N_{surf} is the number of surface nodes, ΔA_i is the surface area element calculated at node i, p_i is the pressure at node i and $p_{i,targ}$ is the target pressure at node i. The optimizer is given the unperturbed geometry and attempts to recover the perturbed wing based on the target surface pressure distribution.

The convergence history for the inverse design case, shown in Figure 4, illustrates that the optimizer successfully recovered the perturbed geometry. The optimality, which is a measure of the gradient, is reduced by 10 orders of magnitude and the objective by 16 orders of magnitude in 40 objective function and gradient evaluations. The solution required 67 hours on 12 processors.

B. Target Pitching Moment

The purpose of this optimization is to minimize the magnitude of the pitching moment for a given target lift coefficient of a rectangular wing with an aspect ratio of 4.0 and a projected area of 2.0 at zero angle of attack. The objective to be minimized is a quadratic penalty function applied to the quarter-chord pitching moment coefficient C_M and is of the form

$$\mathcal{J} = \frac{1}{2} C_M^2. \tag{17}$$

The computational grid is made up of 12 blocks and contains a total of 2.47 million nodes. Each block is fit with a B-spline volume of $9 \times 9 \times 5$ control points. The off-wall distance is 1.12×10^{-6} chord units. The z-coordinates of the B-spline control points are used as design variables, with the exception of the control points near the leading and trailing edges, which are fixed in order to prevent crossover. With the angle of attack fixed at zero, the optimization has 85 design variables. A non-linear inequality constraint is used to maintain a minimum volume equal to that of the original geometry. A non-linear equality constraint is applied to the lift coefficient. The target lift coefficient is set at 0.20, based on the projected area. The flow is fully turbulent at a free-stream Mach number of 0.5 and a Reynolds number of 3 million. The initial geometry has symmetric NACA0012 sections and therefore produces no lift or pitching moment at zero angle of attack.

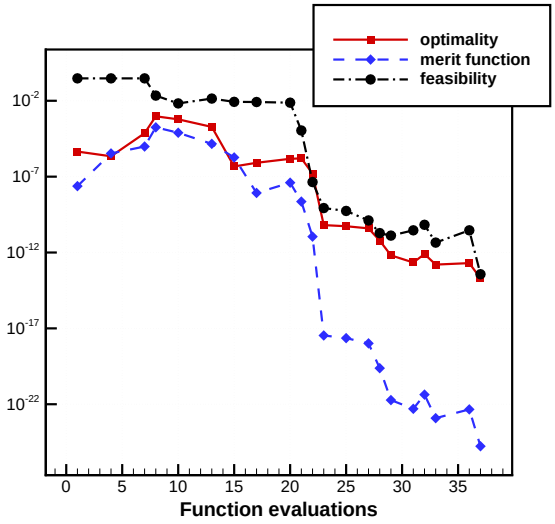


Figure 5: Convergence history for moment penalty optimization

The optimizer produces a final geometry that achieves the target lift coefficient with a quarter-chord pitching moment coefficient of 0.2920×10^{-12}. The convergence history is shown in Figure 5. The optimality is reduced by 8 orders of magnitude in 37 objective function and gradient evaluations. The merit function is reduced by 17 orders of magnitude to 10^{-25}. The optimizer required 100 hours on 12 processors to achieve these results.

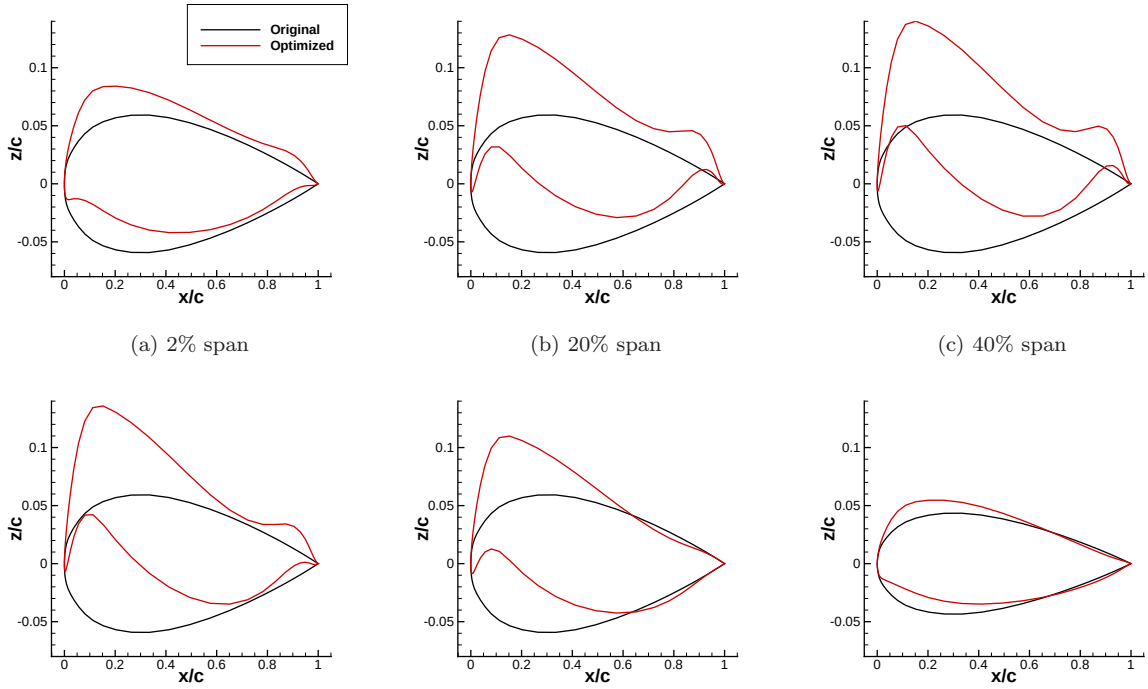
The initial and optimized geometries are compared at six different locations along the span in Figure 6. The pressure coefficient distribution is plotted for each of these sections in Figure 7. In order to reduce the pitching moment to zero, the optimizer has produced reflexed airfoils as expected. This example provides a good test of the algorithm, as the section shape changes are relatively large. As many different geometries can produce finite lift and zero moment at zero angle of attack, the solution is not unique and depends on the initial condition.

C. Lift-Constrained Drag Minimization in Transonic Flow

1. Sweep and Angle of Attack

We aim to optimize the leading edge sweep angle of the ONERA M6 wing to minimize the drag at a fixed lift coefficient of 0.257 in a turbulent transonic flow at a free-stream Mach number of 0.8395, Reynolds number of 11.72 million, and an initial angle of attack of 3.06 degrees. The angle of attack is free in order to meet the lift constraint. The x-coordinates of the B-spline control points are all coupled to the sweep design variable. The initial leading edge sweep angle of the ONERA M6 is 30 degrees. The computational mesh is made up of 24 blocks and a total of 2.187 million nodes. From the grid redistribution technique discussed in Section III, we obtain an off-wall spacing of 7.23×10^{-7} chord units. Each block is parameterized with 9 control points in the stream-wise direction and 5 in the span-wise and off-wall directions. The lift coefficient is constrained to a target value of 0.257 based on a projected area of 1.17.

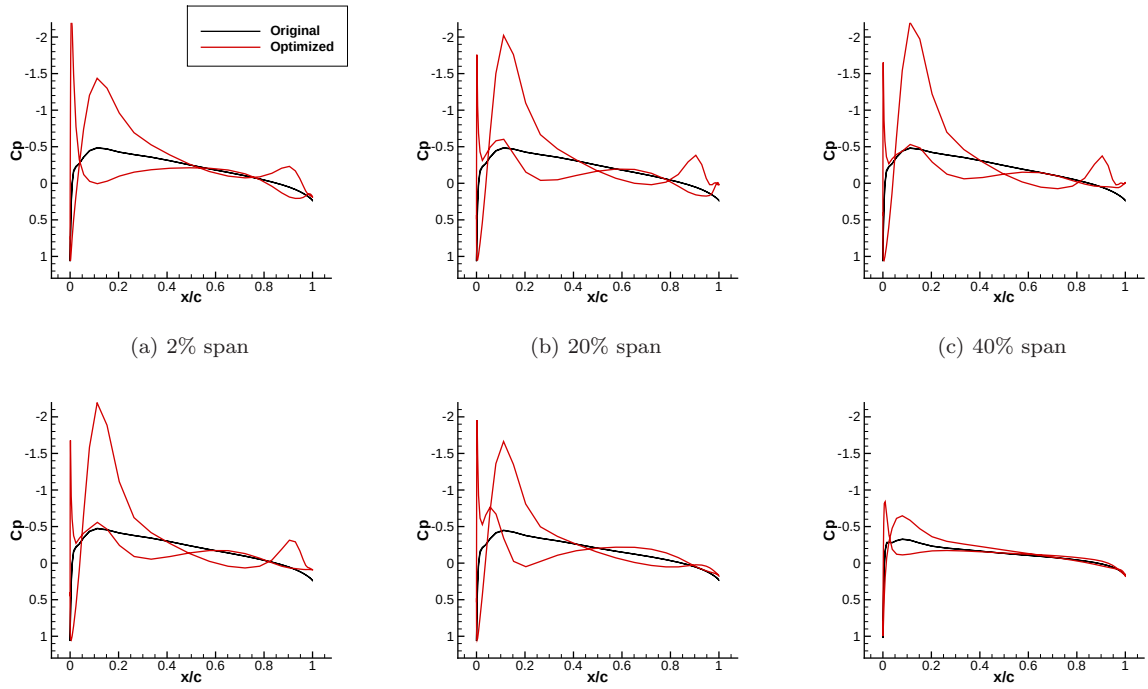
The convergence history for this optimization is shown in Figure 8. The drag coefficient has been reduced by 11%. The optimality has been decreased by 3 orders of magnitude in 37 function evaluations. The wing was swept back by 14.2 degrees, as shown in Figure 9, resulting in a final leading edge sweep angle of 44.2 degrees. The angle of attack was increased to 3.59 degrees. The final result was obtained in 36 hours on 24 processors. This case demonstrates the ability of the mesh movement algorithm to handle large changes in the sweep angle.



(a) 2% span (b) 20% span (c) 40% span

(d) 60% span (e) 80% span (f) 95% span

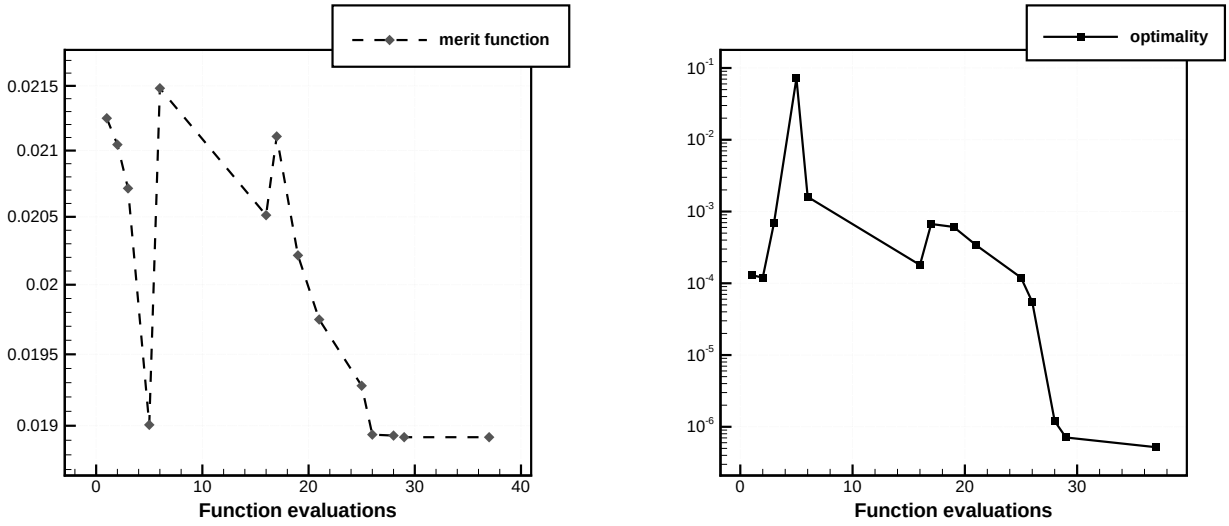
Figure 6: Section shape comparison for moment coefficient optimization



(a) 2% span (b) 20% span (c) 40% span

(d) 60% span (e) 80% span (f) 95% span

Figure 7: Coefficient of pressure for moment coefficient optimization



(a) Merit function

(b) Optimality

Figure 8: Convergence history for transonic optimization with varying sweep

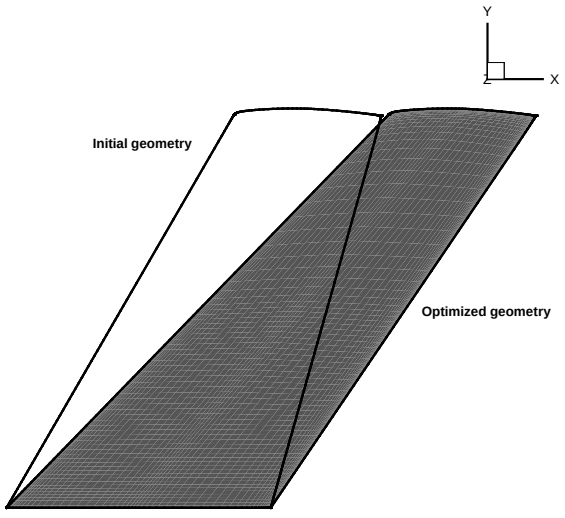
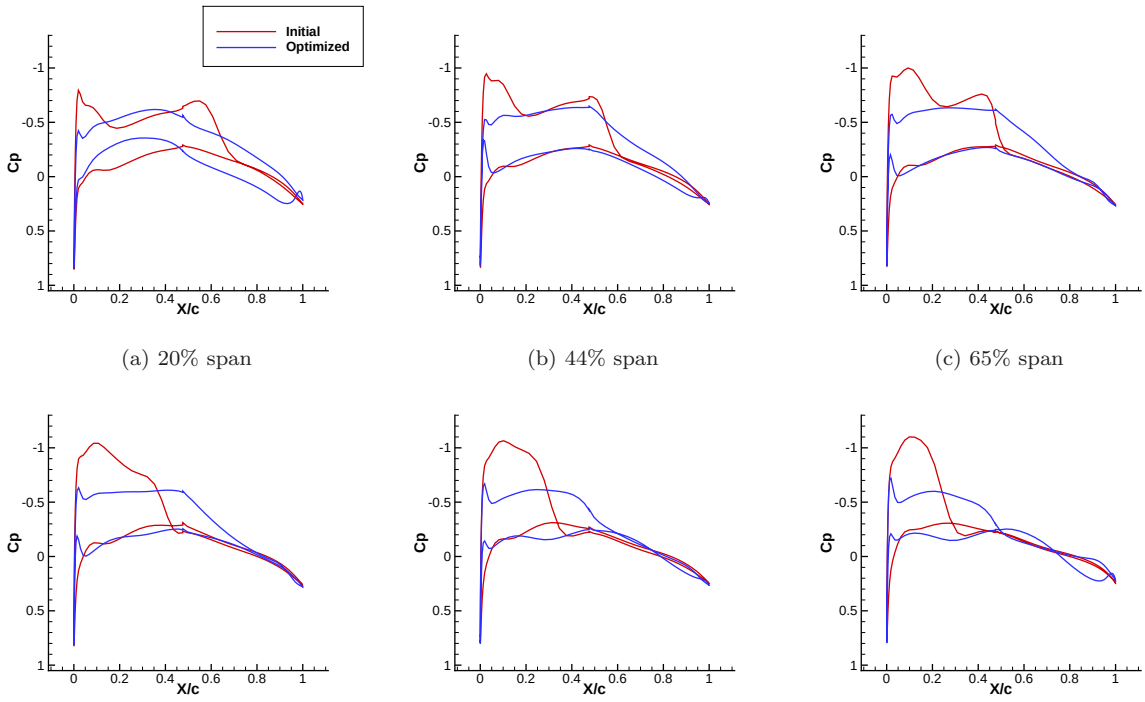


Figure 9: Geometry comparison for transonic optimization with varying sweep



(a) 20% span (b) 44% span (c) 65% span

(d) 80% span (e) 90% span (f) 95% span

Figure 10: Coefficient of pressure for transonic optimization with varying z-coordinates

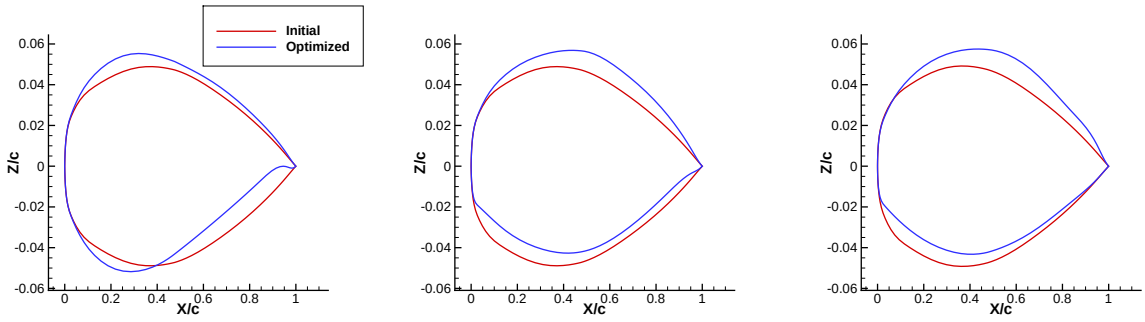
2. *B-spline Control Point z-Coordinates and Angle of Attack*

To test the optimizer's ability to handle more complex cases with more design variables, we aim to minimize the drag of an ONERA M6 wing at a fixed lift coefficient of 0.259 at transonic speed by varying the vertical positions of 225 B-spline control points and the angle of attack. The control points near the leading and trailing edges are fixed to avoid crossover. The free-stream Mach number is 0.8395, the Reynolds number is 11.72 million, and the initial angle of attack is 3.06 degrees. The 24-block, 2.187-million node mesh described in the previous example is used in this case as well. We apply the grid redistribution technique to obtain an off-wall spacing of 1.2×10^{-6}. The control volumes are each made up of $7 \times 7 \times 5$ control points. The lift coefficient, calculated based on the projected area, is constrained to maintain the initial value of 0.259. A minimum volume constraint based on the original geometry is also used.

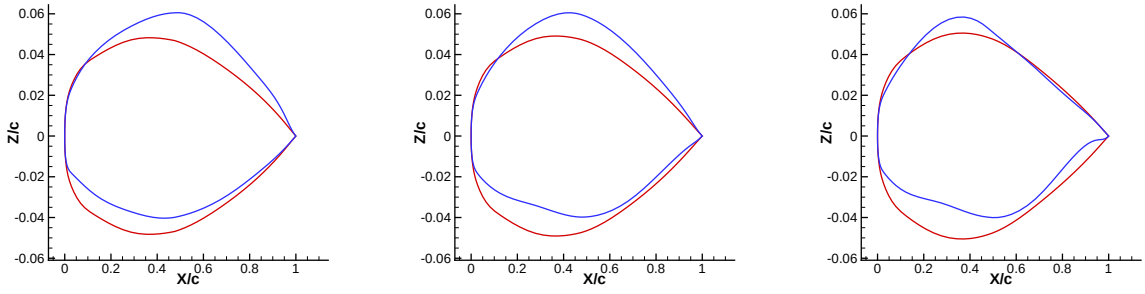
From the plots of the coefficient of pressure in Figure 10, we can see that the initial geometry has two shocks present on the upper surface. Over the course of 72 function evaluations, the optimizer has virtually eliminated the shocks by altering the section shapes and decreasing the angle of attack to 1.91 degrees. The section shape changes are shown in Figure 11. The drag has been reduced by 17.1%. The optimality, shown in the convergence plot in Figure 12, has been reduced by one order of magnitude, but work is underway to obtain deeper convergence.

VII. Conclusions

We have presented an efficient algorithm for aerodynamic shape optimization in the turbulent flow regime. The use of an integrated geometry parameterization and mesh movement scheme balances the robustness of a linear elasticity-based mesh movement algorithm with the speed of an algebraic method. The computational mesh is approximated by a coarse mesh made up of B-spline volumes. The nodes of this mesh are the B-spline control points and are moved using a method based on the principles of linear elasticity; subsequently, the fine mesh is updated based on the B-spline volume basis functions. Because the size of the control mesh is relatively small compared to the fine one, mesh movement and the evaluation of the mesh adjoint

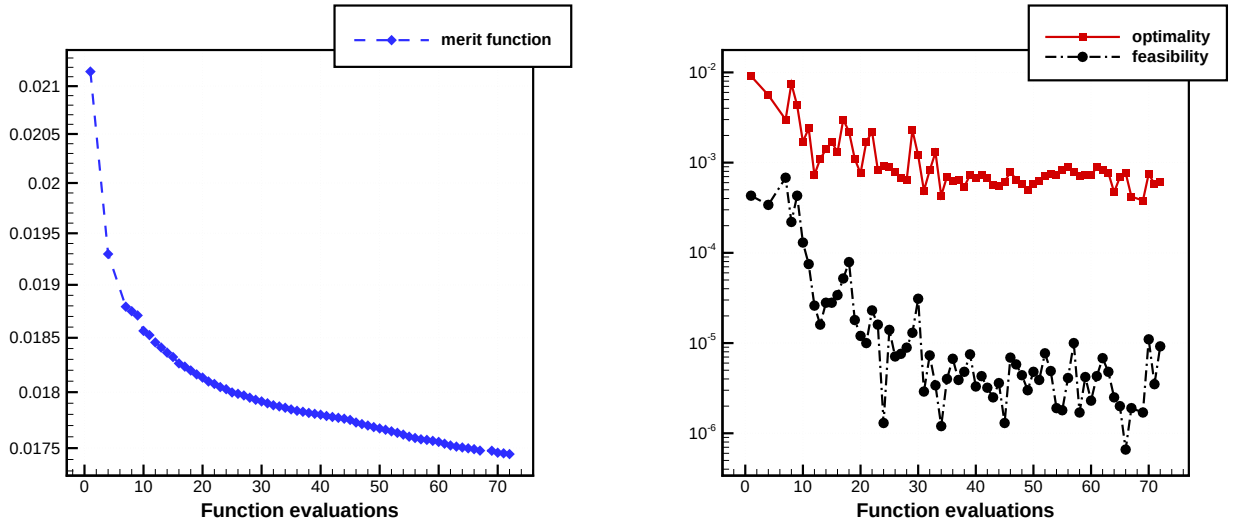


(a) 20% span (b) 44% span (c) 65% span



(d) 80% span (e) 90% span (f) 95% span

Figure 11: Section shape comparison for transonic optimization with varying z-coordinates



(a) Merit function (b) Optimality

Figure 12: Convergence history for transonic optimization with varying z-coordinates

equations can be executed at a low computational cost. Grid refinement and grid redistribution strategies are discussed for obtaining meshes suitable for turbulent flows from coarser meshes in order to address difficulties arising in fitting B-spline volumes to finely spaced meshes. A Newton-Krylov flow solver with approximate-Schur preconditioning is used for flow analysis. We apply the discrete-adjoint method to the evaluation of the gradient. The SQP optimizer SNOPT is used to perform constrained optimizations. We have successfully tested the algorithm using an inverse design case based on surface pressure and further demonstrated the performance of the algorithm through the design of a wing with a zero pitching moment and two lift-constrained drag minimizations in transonic flow. The algorithm is able to exhibit excellent convergence when dealing with large shape changes. The results show the overall methodology to be an efficient and reliable approach to aerodynamic shape optimization. Future work will include planar and non-planar designs in subsonic and transonic flow, as well as multi-point optimizations, and extension to aerostructural optimization.

Acknowledgments

The authors gratefully acknowledge financial assistance from the Natural Sciences and Engineering Research Council (NSERC), the Canada Research Chairs program, Bombardier Aerospace, MITACS, and the University of Toronto.

References

[1]Holland, J. H., *Adaptation in Natural and Artificial Systems*, The University of Michigan Press, Ann Arbor, Michigan, 1975.

[2]Nocedal, J. and Wright, S. J., *Numerical Optimization*, Springer–Verlag, Berlin, Germany, 1999.

[3]Gill, P. E., Murray, W., and Saunders, M. A., "SNOPT: an SQP algorithm for large-scale constrained optimization," *SIAM Journal on Optimization*, Vol. 12, No. 4, 2002, pp. 979–1006.

[4]Fletcher, R. and Leyffer, S., "Nonlinear programming without a penalty function," *Mathematical Programming*, Vol. 91, No. 2, Jan. 2002, pp. 239–269.

[5]Byrd, R. H., Gould, N. I., Nocedal, J., and Waltz, R. A., "An algorithm for nonlinear optimization using linear programming and equality constrained subproblems," *Mathematical Programming*, Vol. 100, No. 1, May 2004, pp. 27–48.

[6]Chernukhin, O. and Zingg, D. W., "An investigation of multi-modality in aerodynamic shape optimization," *20th AIAA Computational Fluid Dynamics Conference*, No. AIAA–2011–3070, Honolulu, Hawaii, United States, June 2011.

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

[8]Pironneau, O., "On optimum design in fluid mechanics," *Journal of Fluid Mechanics*, Vol. 64, No. 1, 1974, pp. 97–110.

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

[10]Jameson, A. and Reuther, J., "Control theory based airfoil design using Euler equations," *AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, Panama City Beach, Sept. 1994.

[11]Reuther, J., Jameson, A., Farmer, J., Martinelli, L., and Saunders, D., "Aerodynamic shape optimization of complex aircraft configurations via an adjoint formulation," *The 34rd AIAA Aerospace Sciences Meeting and Exhibit*, No. AIAA–1996–0094, Reno, Nevada, 1996.

[12]Reuther, J. J., Jameson, A., Alonso, J. J., Rimlinger, M. J., and Saunders, D., "Constrained multipoint aerodynamic shape optimization using an adjoint formulation and parallel computers, part 1," *AIAA Journal*, Vol. 36, No. 1, Jan. 1999, pp. 51–60.

[13]Reuther, J. J., Jameson, A., Alonso, J. J., Rimlinger, M. J., and Saunders, D., "Constrained multipoint aerodynamic shape optimization using an adjoint formulation and parallel computers, part 2," *AIAA Journal*, Vol. 36, No. 1, Jan. 1999, pp. 61–74.

[14]Jameson, A., Martinelli, L., and Pierce, N. A., "Optimum aerodynamic design using the Navier-Stokes equations," *Theoretical and Computational Fluid Dynamics*, Vol. 10, 1998, pp. 213–237.

[15]Anderson, W. K. and Bonhaus, D. L., "Airfoil design on unstructured grids for turbulent flows," *AIAA Journal*, Vol. 37, No. 2, Feb. 1999, pp. 185–191.

[16]Nemec, M. and Zingg, D. W., "Newton-Krylov algorithm for aerodynamic design using the Navier-Stokes equations," *AIAA Journal*, Vol. 40, No. 6, 2002, pp. 1146–1154.

[17]Nemec, M., Zingg, D. W., and Pulliam, T. H., "Multipoint and multi-objective aerodynamic shape optimization," *AIAA Journal*, Vol. 42, No. 6, 2004, pp. 1057–1065.

[18]Driver, J. and Zingg, D. W., "Numerical aerodynamic optimization incorporating laminar-turbulent transition prediction," *AIAA Journal*, Vol. 45, No. 8, Aug. 2007, pp. 1810–1818.

[19]Hicken, J. E. and Zingg, D. W., "Aerodynamic optimization algorithm with integrated geometry parameterization and mesh movement," *AIAA Journal*, Vol. 48, No. 2, Feb. 2010, pp. 400–413.

[20]Hicken, J. E. and Zingg, D. W., "A parallel Newton-Krylov solver for the Euler equations discretized using simultaneous approximation terms," *AIAA Journal*, Vol. 46, No. 11, Nov. 2008, pp. 2773–2786.

[21]Hicken, J. E. and Zingg, D. W., "Induced drag minimization of nonplanar geometries based on the Euler equations," *AIAA Journal*, Vol. 48, No. 11, 2010, pp. 2564–2575.

[22]Truong, A. H., Oldfield, C. A., and Zingg, D. W., "Mesh movement for a discrete-adjoint Newton-Krylov algorithm for aerodynamic optimization," *AIAA Journal*, Vol. 46, No. 7, July 2008, pp. 1695–1704.

[23]Osusky, M., Hicken, J. E., and Zingg, D. W., "A parallel Newton-Krylov-Schur flow solver for the Navier-Stokes equations using the SBP-SAT approach," *48th AIAA Aerospace Sciences Meeting and Aerospace Exposition*, No. AIAA–2010–116, Orlando, Florida, United States, Jan. 2010.

[24]Spalart, P. R. and Allmaras, S. R., "A one-equation turbulence model for aerodynamic flows," No. 92–0439, Jan. 1992.

[25]Svärd, M., Mattsson, K., and Nordström, J., "Steady-state computations using Summation-by-Parts operators," *Journal of Scientific Computing*, Vol. 24, No. 1, July 2005, pp. 79–95.

[26]Carpenter, M. H., Gottlieb, D., and Abarbanel, S., "Time-stable boundary conditions for finite-difference schemes solving hyperbolic systems: methodology and application to high-order compact schemes," *Journal of Computational Physics*, Vol. 111, No. 2, 1994, pp. 220–236.

[27]Hesthaven, J. S., "A stable penalty method for the compressible Navier-Stokes equations: III. Multidimensional domain decomposition schemes," *SIAM Journal on Scientific Computing*, Vol. 20, No. 1, 1998, pp. 62–93.

[28]Carpenter, M. H., Nordström, J., and Gottlieb, D., "A stable and conservative interface treatment of arbitrary spatial accuracy," *Journal of Computational Physics*, Vol. 148, No. 2, 1999, pp. 341–365.

[29]Nordström, J. and Carpenter, M. H., "High-order finite difference methods, multidimensional linear problems, and curvilinear coordinates," *Journal of Computational Physics*, Vol. 173, No. 1, 2001, pp. 149–174.

[30]Pironneau, O., *Optimal shape design for elliptic systems*, Springer-Verlag, 1983.

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

[32]Hicken, J. E. and Zingg, D. W., "A simplified and flexible variant of GCROT," *SIAM Journal on Scientific Computing*, Vol. 32, No. 3, 2010, pp. 1672–1694.