Available online at www.sciencedirect.com
ScienceDirect
Journal of Computational Physics 227 (2008) 2724–2742
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

Adjoint sensitivity computations for an embedded-boundary Cartesian mesh method

Marian Nemec [a,b,*], Michael J. Aftosmis [b]

[a] ELORET Corporation, Sunnyvale, CA 94086, USA
[b] NASA Ames Research Center, MS T27B, Moffett Field, CA 94035, USA

Received 14 May 2007; received in revised form 10 October 2007; accepted 9 November 2007
Available online 22 November 2007

Abstract

We present a new approach for the computation of shape sensitivities using the discrete adjoint and flow-sensitivity methods on Cartesian meshes with general polyhedral cells (cut-cells) at the wall boundaries. By directly linearizing geometric constructors of the cut-cells, an efficient and robust computation of shape sensitivities is achieved for problems governed by the Euler equations. The accuracy of the linearization is verified by the use of a model problem with an exact solution. Verification studies show that the convergence rate of gradients is second-order for design variables that do not alter the boundary shape, and is reduced to first-order for shape design problems. The approach is applied to several three-dimensional problems, including inverse design and shape optimization of a re-entry capsule in hypersonic flow. The results show that reliable approximations of the gradient are obtained in all cases. The approach is well-suited for geometry control via computer-aided design, and is especially effective for conceptual design studies with complex geometry where fast turn-around time is required.

Keywords: Aerodynamic shape optimization; Cartesian mesh; Adjoint

1. Introduction

An important factor in the application of numerical optimization as a practical tool of aerodynamic design is the need to handle complex geometry problems in an automatic fashion. Each iteration of a typical optimization procedure requires a new analysis of the evolving geometry. This involves the sequence of re-meshing the computational domain and re-solving the flow equations. In this setting, automatic mesh generation is essential. Most approaches rely on interactive mesh generation to create an initial mesh, and then use a mesh-perturbation scheme to deform that mesh as the surface geometry changes. A promising alternative is offered by embedded-boundary Cartesian mesh methods [8,2,9]. These methods provide fast and robust mesh generation that is insensitive to the complexity of the surface shape. In this work, we combine the automation

* Corresponding author. Address: NASA Ames Research Center, MS T27B, Moffett Field, CA 94035, USA.
E-mail address: Marian.Nemec@nasa.gov (M. Nemec).

0021-9991/$ - see front matter © 2007 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2007.11.018

capabilities of Cartesian methods with efficient approaches for sensitivity analysis to accelerate the convergence of the optimization procedure.

Many examples are available in the literature that demonstrate the effectiveness of Cartesian methods for a variety of optimization algorithms. Cliff et al. [7] use gradient methods, Rodriguez [37] uses a simplex method, Keane [22] proposes the use of response surfaces, and Nelson et al. [29] investigate the use of genetic algorithms. In these studies, the computation of the objective function gradient, or sensitivity analysis, was either not required or performed with finite-difference approximations. As the complexity of optimization problems increases, for example by considering more design variables, objective functions and constraints, the dominant issue becomes computational cost. Efficient techniques for reducing the computational cost are the adjoint and flow-sensitivity methods [19,4,27]. The cost of gradient computations using the adjoint method is essentially independent of the number of design variables. Conversely, the flow-sensitivity method is attractive for problems with many objectives or flow-dependent constraints. The resulting sensitivity information is useful for not only gradient-based methods, but also convergence acceleration of response surface methods and genetic algorithms.

The formulation of the adjoint and flow-sensitivity methods requires a complete linearization of the governing equations with respect to the design variables. This includes the objective function, flow equations, and the relations used to define the boundary shape. Detailed overviews are presented by Jameson [20] and Giles and Pierce [15]. The most relevant aspect in the present context is the effect of boundary shape variations in the linearization of the flow equations. This typically involves the linearization of a mesh-perturbation scheme, such as spring-analogy or linear-elasticity methods on unstructured meshes [33,43,13] and algebraic node-redistribution functions on structured meshes [36,23]. The linearization introduces an additional adjoint equation in the optimization procedure [35,26,41]; however, the mesh-perturbation scheme provides a smooth mapping that reduces the influence of truncation errors on the gradient computation. While this approach is well understood for implementations in conjunction with body-fitted meshes, a different approach is required for non-body-fitted Cartesian meshes. This is due to the layer of irregular cells, or cut-cells, adjacent to the boundary, that changes arbitrarily for each instance of the geometry. Similar problems also arise for shape topology changes and implementations in conjunction with solution-based mesh adaptation.

One approach is to avoid mesh-perturbation schemes altogether. A good example is the Cartesian TRANAIR code [44], which is based on the full-potential equation with viscous corrections. In this approach, Huffman et al. [18] developed and linearized a transpiration boundary condition to approximate the effects of the changing boundary surface. They obtained good gradient accuracy for problems where the design variables involve shape changes normal to the surface. The optimization procedure involves the solution of a sequence of subproblems with a periodic re-meshing of the geometry. More recently, Duvigneau and Pelletier [12] introduced a continuous formulation that uses higher-order Taylor series approximations near the boundary to obtain accurate shape sensitivities for two-dimensional problems.

A different strategy is motivated by studies on unstructured meshes that investigate the extent of mesh perturbations (in the wall-normal direction) required for accurate gradients. Anderson and Venkatakrishnan [3] and Lu [24] restricted the mesh perturbations to just the surface boundary cells, with all interior cells held fixed. The results show that for local shape deformations, away from singularities, reliable approximations of the gradient can be obtained. A similar approach was presented by Dadone and Grossman [10] on Cartesian meshes for the two-dimensional Euler equations. This work used an immersed-boundary methodology to apply finite-difference approximations for shape sensitivities in the boundary intersecting cells. Perturbations of only the boundary cells were also considered by Jameson and Kim [21], who introduced a reduced gradient approach for the continuous adjoint formulation on structured grids. Their results indicate that for shape optimization governed by the Euler equations, the reduced approach is accurate and provides significant computational savings. Similar results are also presented by Soto and Löhner [39] on unstructured meshes.

In this work, we propose an automatic and efficient method for the computation of shape sensitivities on non-body-fitted Cartesian meshes. Our approach is similar to a mesh-perturbation scheme where the perturbations are restricted to the boundary cells. The key difference is that we maintain the orientation of the Cartesian-aligned faces of the cut-cells as the boundary surface evolves. The linearization of the geometric terms at the surface boundary is obtained directly from the geometry constructors of the mesh generator. In the linearization, no assumptions are made regarding the connectivity of the surface triangulation with the volume

mesh. As a result, our approach offers a flexible interface for geometry manipulation, which may include parametric computer-aided design (CAD) and in-house geometry modelers. Furthermore, the new approach is inexpensive, since it does not require the solution of an additional adjoint equation, and it may be applicable to a broader range of problems than schemes based on the transpiration boundary condition.

To assess the accuracy of the computed gradients, we present detailed verification studies using comparisons with analytic model problems and finite-difference approximations. Thereafter, we investigate the performance of the new method on two representative shape-optimization problems, namely, inverse design and an enhancement of the lift-to-drag ratio for an Apollo-like re-entry capsule in hypersonic flow. Factors under consideration include robustness of the adjoint and flow-sensitivity solvers, and efficiency of the gradient computation. In addition, the design examples are also used to highlight the capability of the method to use various geometry modeling and manipulation tools, including a direct-CAD interface for the capsule problem and an analytic B-spline approach for the inverse design problem.

2. Optimization problem

The aerodynamic optimization problem we consider in this work consists of determining values of design variables, X, that minimize a given objective function

$$\min_X \quad \mathcal{J}(X, Q) \tag{1}$$

where $\mathcal{J}$ represents a scalar objective function defined by a surface integral, for example lift or drag, and $Q = [\rho, \rho u, \rho v, \rho w, \rho E]^{\mathrm{T}}$ are the continuous flow variables. The variables are forced to satisfy the flow equations within a feasible region of the design space Ω

$$\mathcal{F}(X, Q) = 0 \quad \forall X \in \Omega \tag{2}$$

which implicitly defines $Q = f(X)$. We use a discrete formulation to compute the objective function gradient $\mathrm{d}\mathcal{J}/\mathrm{d}X$, where the governing equations, Eqs. (1) and (2), are first discretized and then differentiated. In the following section, we present background information on the Cartesian mesh generator and the flow solution method to help anchor the subsequent discussion on linearization.

3. Flow equations and numerical method

The governing flow equations are the three-dimensional Euler equations of a perfect gas. For a finite region of space with volume V and surface area S, the equations are given by

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_V Q\,\mathrm{d}V + \oint_S F \cdot \hat{\mathbf{n}}\,\mathrm{d}S = 0 \tag{3}$$

where F is the inviscid flux tensor and $\hat{\mathbf{n}}$ is the outward facing unit normal vector.[1] The equations are discretized on a multilevel Cartesian mesh with embedded boundaries. The mesh consists of regular Cartesian hexahedra everywhere, except for a layer of body-intersecting cells, or *cut-cells*, adjacent to the body surface. As illustrated in Figs. 1 and 2, these cells are polyhedra of arbitrary degree. Spatial discretization uses a cell-centered approach, where the control volumes correspond to the mesh cells and the cell-averaged value of Q, denoted by $\overline{Q}$, is located at the centroid of each cell. We consider steady-state problems and write the resulting discrete system of equations as

$$\vec{R}(\vec{Q}, \vec{M}, X) = 0 \tag{4}$$

where $\vec{Q} = [\overline{Q}_1, \overline{Q}_2, \ldots, \overline{Q}_N]^{\mathrm{T}}$ is the discrete solution vector for all N cells of a given mesh $\vec{M}$. The flux residual in each cell i is expressed as

$$R_i = \sum_{j \in V_i} (H \cdot \hat{\mathbf{n}} S)_j \tag{5}$$

[1] Bold type denotes Cartesian vectors.

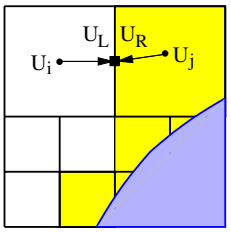


Fig. 1. Multilevel Cartesian mesh in two-dimensions with a cut-cell boundary.

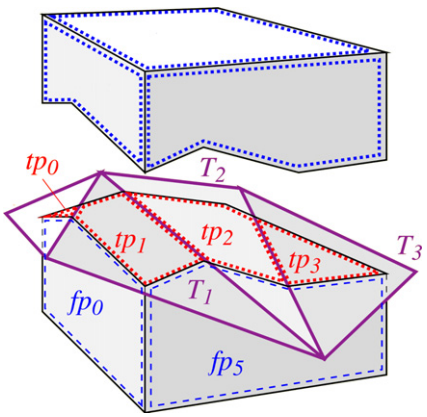


Fig. 2. Cut-cell geometry for surface triangulation (T) with triangle–polygons (tp) and face-polygons (fp).

where j denotes the faces of control volume V_i with area S, and H represents the numerical flux function. The flux function is evaluated at the face centroids using the flux–vector splitting approach of van Leer [42]. Primitive variables, $U = [\rho, u, v, w, p]^{\mathrm{T}}$, are used for the reconstruction of the solution to the common face between adjacent cells, such that $H = H(U_{\mathrm{L}}, U_{\mathrm{R}})$. This is illustrated in Fig. 1 for two neighboring Cartesian cells. The left state is given by

$$U_{\mathrm{L}} = \overline{U}_i + \mathbf{d}_{\mathrm{L}} \phi_i \nabla U_i \tag{6}$$

where $\mathbf{d}_{\mathrm{L}}$ is a vector from the cell centroid to the face centroid, ϕ is a matrix of slope limiter values used to ensure monotonic reconstructions, and ∇U is the solution gradient determined via a linear least-squares procedure. A similar expression is used for the right state. The wall-boundary conditions are enforced weakly using pressure from interior cells in the momentum equations.

The evaluation of mesh dependent terms in Eqs. (5) and (6), such as areas, normals and centroids, is straightforward for regular Cartesian cells. Our focus is on the cut-cells, because an infinitesimal perturbation of the boundary shape affects only these cells in the mesh. The cut-cell geometry is defined by the intersection of the surface triangulation with the faces of Cartesian hexahedra. Fig. 2 labels two main geometry features in a generic cut-cell, namely, triangle- and face-polygons. Triangle-polygons are formed from the portions of triangles that lie within the cell. Face-polygons are formed from the Cartesian aligned faces of the cell after being trimmed by the triangulation. A specific example is shown in Fig. 3, where a Cartesian hexahedron is split into two cut-cells by the surface triangulation. We describe the construction of the polygon $(\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D})$ for triangle $(\mathbf{V}_0, \mathbf{V}_1, \mathbf{V}_2)$. We require the location of intersection points that lie on Cartesian edges, e.g. point $\mathbf{A}$, and also those that lie on triangle edges, e.g. point $\mathbf{D}$. Focusing on point $\mathbf{D}$, its location along the triangle edge $\mathbf{V}_0\mathbf{V}_1$ is given by

$$\mathbf{D} = \mathbf{V}_0 + s(\mathbf{V}_1 - \mathbf{V}_0) \tag{7}$$

where s denotes the distance fraction of the face location relative to the vertices $\mathbf{V}_0$ and $\mathbf{V}_1$. A similar constructor is used for points along Cartesian edges that pierce the surface triangulation, such as point $\mathbf{A}$ [30]. The

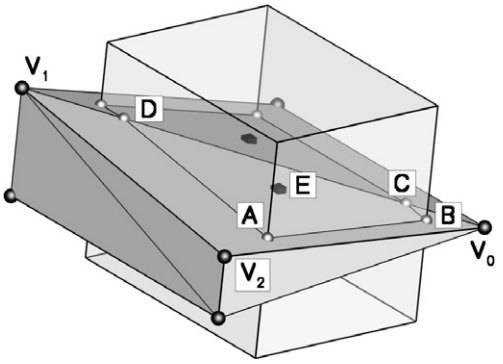


Fig. 3. Intersection of a Cartesian hexahedron with triangle $(\mathbf{V}_0, \mathbf{V}_1, \mathbf{V}_2)$ creates pierce points $(\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D})$ that form a triangle–polygon with a surface centroid **E**.

computation of areas and centroids for the face-polygons is accomplished by subdividing each face into triangle "fans" that are defined by the pierce points and the corners of the hexahedron. The computation of volume centroids uses the divergence theorem and involves integrating over the triangle- and face-polygons of the cut-cell.

The geometric information required for the implementation of wall boundary conditions consists of the wall normal, the area of the surface patch within the cut-cell and the surface centroid. We consider two approaches for the representation of the wall boundary. In the first approach, the computation of the wall normal is based on a planar approximation to the variation of the triangulation within each cell. This agglomerated normal vector is computed by requiring the geometric closure of each cut-cell

$$\sum_{j \in V_i} S_j \cdot \hat{\mathbf{n}}_j = 0 \tag{8}$$

where the sum is performed over the faces j of the cell. In the second approach, the boundary conditions are enforced for each triangle–polygon of the cut-cell, which provides a "sub-cell" description of the boundary. Referring to Fig. 3, the flow solution is reconstructed to the triangle–polygon centroid, point **E**, and the wall unit normal is inherited from the parent triangle.

Steady-state flow solutions are obtained using a five-stage Runge–Kutta scheme with local time stepping, multigrid, and a domain decomposition scheme for parallel computing. For further details on the spatial discretization and flow solution, see Aftosmis et al. [2,1] and Berger et al. [5].

4. Gradient computation

The design variables that appear directly in Eq. (4) involve parameters that do not change the computational domain, such as the Mach number, angle of attack, and side-slip angle. The influence of shape design variables on the residuals in Eq. (4) is implicit via the computational mesh $\vec{M}$:

$$\vec{M} = f[\vec{T}(X)] \tag{9}$$

where $\vec{T}$ denotes a triangulation of the wetted surface. In the following sections, we present the linearization of the residual equations and the objective function to obtain the gradient.

4.1. Discrete adjoint and flow-sensitivity methods

The gradient of the discrete objective function $\mathcal{J}(X, \vec{M}, \vec{Q})$ with respect to a design variable X is given by

$$\frac{\mathrm{d}\mathcal{J}}{\mathrm{d}X} = \frac{\partial \mathcal{J}}{\partial X} + \frac{\partial \mathcal{J}}{\partial \vec{M}} \frac{\partial \vec{M}}{\partial \vec{T}} \frac{\partial \vec{T}}{\partial X} + \frac{\partial \mathcal{J}}{\partial \vec{Q}} \frac{\mathrm{d}\vec{Q}}{\mathrm{d}X} \tag{10}$$

where both $\partial\mathcal{J}/\partial\vec{M}$ and $\partial\mathcal{J}/\partial\vec{Q}$ are row vectors. The evaluation of the term $\mathrm{d}\vec{Q}/\mathrm{d}X$, referred to as the flow sensitivities, is obtained by combining Eqs. (4) and (9), and differentiating with respect to the design variables

$$\frac{\partial\vec{R}}{\partial\vec{Q}}\frac{\mathrm{d}\vec{Q}}{\mathrm{d}X} = -\left(\frac{\partial\vec{R}}{\partial X} + \frac{\partial\vec{R}}{\partial\vec{M}}\frac{\partial\vec{M}}{\partial\vec{T}}\frac{\partial\vec{T}}{\partial X}\right) \tag{11}$$

We assume that the implicit function $\vec{Q}(X)$ is sufficiently smooth, and note that $\mathrm{d}\vec{R}/\mathrm{d}X = 0$ because Eq. (2) holds for any design variable. The direct, or *flow-sensitivity*, method results from solving Eq. (11) for the flow sensitivities $\mathrm{d}\vec{Q}/\mathrm{d}X$ and using these values in Eq. (10) to obtain the gradient.

The *adjoint* equation is obtained by combining Eqs. (11) and (10) and defining the following intermediate problem:

$$\frac{\partial\vec{R}}{\partial\vec{Q}}^{\mathrm{T}}\vec{\psi} = \frac{\partial\mathcal{J}}{\partial\vec{Q}}^{\mathrm{T}} \tag{12}$$

where the vector $\vec{\psi}$ represents the adjoint variables. The corresponding expression for the gradient is given by

$$\frac{\mathrm{d}\mathcal{J}}{\mathrm{d}X} = \frac{\partial\mathcal{J}}{\partial X} + \underbrace{\frac{\partial\mathcal{J}}{\partial\vec{M}}\frac{\partial\vec{M}}{\partial\vec{T}}\frac{\partial\vec{T}}{\partial X}}_{\mathrm{A}} - \vec{\psi}^{\mathrm{T}}\left(\frac{\partial\vec{R}}{\partial X} + \underbrace{\frac{\partial\vec{R}}{\partial\vec{M}}\frac{\partial\vec{M}}{\partial\vec{T}}\frac{\partial\vec{T}}{\partial X}}_{\mathrm{B}}\right) \tag{13}$$

The adjoint method is advantageous for problems with many design variables and few objective functions, because Eq. (12) is independent of the design variables.

The solution algorithm for the flow-sensitivity and adjoint equations leverages the Runge–Kutta time-marching scheme and the parallel multigrid method of the flow solver. The algorithm is implemented using the duality-preserving approach [14], such that the asymptotic convergence rate of the flow, flow-sensitivity, and adjoint solvers should be identical. The matrix–vector products associated with the flow-Jacobian matrix, left side of Eqs. (11) and (12), are computed on-the-fly using a two-pass strategy over the faces of the mesh. This ensures that the memory usage of the flow-sensitivity and adjoint solvers is equivalent to that of the flow solver.

The flow-Jacobian matrix, as well as the term $\partial\mathcal{J}/\partial\vec{Q}$ in Eq. (12), are derived by hand. The limiter function is treated as a constant in the linearization. We find that the effect of this simplification is small on gradient accuracy and convergence characteristics of the adjoint and flow-sensitivity solvers in transonic and low supersonic flow [32,31]. We discuss this issue further in the Results section for supersonic and hypersonic flows. The discretization and linearization of the objective function is consistent with the implementation of the wall-boundary condition in the flow equations, Eq. (4). We use the reconstructed pressure value in conjunction with either the agglomerated surface normal or sub-cell geometry to evaluate the objective function.

In terms of CPU time per iteration, the cost of the adjoint and flow-sensitivity solvers is about 15% greater than the flow solver. This is due to the repeated evaluations of the matrix–vector products, which involve more arithmetic operations relative to the flow solver, but some computational savings are obtained by pre-computing and storing the limiter and time-step values [31,30]. In practice, it is sufficient to reduce the residuals of the adjoint and flow-sensitivity equations by four orders of magnitude to obtain reliable gradients, and consequently, the overall cost of solving these equations is usually less than a flow solution.

The computation of partial derivative terms $\partial\mathcal{J}/\partial X$ and $\partial\vec{R}/\partial X$ in Eq. (13) is straightforward, because these terms do not involve derivatives of the surface shape. The remaining partial derivative terms in Eq. (13), labeled as A and B, represent the differentiation of the objective function and residual equations with respect to design variables that alter the surface shape. We discuss the computation of these terms in the next section.

4.2. Computation of shape sensitivities

An important feature of the cut-cell Cartesian approach is that there is no prescribed connectivity between the surface triangulation and the volume mesh. This feature separates the tasks of geometry modeling and surface triangulation from those of volume mesh generation. Moreover, the computation of shape sensitivities

associated with the vertices of the surface triangulation, $\partial\vec{T}/\partial X$ in Eq. (13), is decoupled from the volume mesh sensitivities, $\partial\vec{M}/\partial\vec{T}$, which include the linearization of the Cartesian-face areas and centroids, volume centroids and wall normals. Consider the linearization of the residual equations (Eq. (5)):

$$\frac{\partial R_i}{\partial X} = \sum_{j\in V_i}\left[H\cdot\frac{\partial(\hat{\mathbf{n}}S)}{\partial X} + \frac{\partial H}{\partial X}\cdot\hat{\mathbf{n}}S\right]_j \tag{14}$$

where the linearization of the flux function involves the reconstruction of the flow solution to the cell face. For example, the linearization of Eq. (6) for the left state is given by

$$\frac{\partial U_{\mathrm{L}}}{\partial X} = \phi_i\left(\mathbf{d}_{\mathrm{L}}\frac{\partial\nabla U_i}{\partial X} + \nabla U_i\frac{\partial\mathbf{d}_{\mathrm{L}}}{\partial X}\right) \tag{15}$$

The mesh sensitivities required in the evaluation of Eqs. (14) and (15), such as $\partial\mathbf{d}_{\mathrm{L}}/\partial X$ and $\partial\hat{\mathbf{n}}/\partial X$, are obtained from the linearization of the mesh generator. These terms are non-zero only in the cut-cells and the crux in their evaluation is the linearization of the pierce points that form the triangle- and face-polygons described in Section 3. Referring to Eq. (7), this linearization is given by

$$\frac{\partial\mathbf{D}}{\partial X} = \frac{\partial\mathbf{V}_0}{\partial X} + s\left(\frac{\partial\mathbf{V}_1}{\partial X} - \frac{\partial\mathbf{V}_0}{\partial X}\right) + (\mathbf{V}_1 - \mathbf{V}_0)\frac{\partial s}{\partial X} \tag{16}$$

An advantage of this formulation is that the dependence of pierce-point sensitivities on the surface triangulation in Eq. (16) is determined on-the-fly for each instance of the surface geometry. Put another way, there is no requirement for a one-to-one triangle mapping as the surface geometry changes.

Fig. 4 shows an example linearization of the face centroid locations for the Cartesian faces of a cut-cell. The surface triangulation contains a single vertex ($\mathbf{V}_1$) with a non-zero shape sensitivity, denoted by a dashed vector. The resulting sensitivities of the face centroids are denoted by solid vectors. Note that "motion" of the face centroids is constrained to the plane of the face. This contrasts with mesh-perturbation approaches for body-fitted meshes, where the cell faces are bound to the vertices of the triangulation and hence are "convected" with the vertices as they move.

The sensitivity of agglomerated wall normals is obtained via a direct linearization of Eq. (8). This is based on the observation that the geometric closure of a cut-cell should be satisfied for any shape perturbation. The flow-solution gradient, ∇U, in Eq. (15) involves the linearization of the least-squares procedure based on Cholesky factorization. As a result, a perturbation of the surface shape influences the residual sensitivities not only in the cut-cells, but also in their first and second nearest-neighbors. Overall, the linearization of the residual equations is computed in parallel for each design variable and reuses the data structures of the mesh generator and flow solver. The CPU time requirements are minimal. The evaluation of mesh sensitivities involves simple algebraic expressions, e.g. Eq. (16), and the residual sensitivity vector, Eq. (14), is non-zero

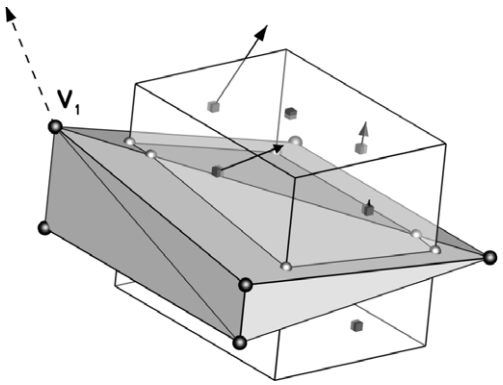


Fig. 4. Example sensitivity of face centroid locations (solid vectors) to a shape perturbation of vertex $\mathbf{V}_1$ of the surface triangulation (dashed vector). Sensitivities are lengthened by a factor of three for visualization.

only in the near-body cells. Hence, in a domain with $\mathcal{O}(N^3)$ cells, the work associated with the computation of residual sensitivities is proportional to $\mathcal{O}(N^2)$.

5. Verification studies

We present two verification studies for the adjoint and flow-sensitivity methods. In the first problem, we investigate the convergence of the gradient error for design variables that do not alter the boundary shape. This establishes a benchmark for the second problem, where we examine the accuracy of gradients for shape design variables. In both problems, we use second-order accurate spatial discretization without limiters, i.e., the slope limiter in Eq. (6) is unity. The flow, flow-sensitivity, and adjoint equations are converged 12 orders of magnitude and the sub-cell wall boundary conditions are used.

5.1. Supersonic vortex model problem

To verify the implementation of the adjoint and flow-sensitivity methods, we investigate the error convergence rate of a representative objective function and its gradient on a model problem with a known analytic solution. The problem involves isentropic flow between concentric circular arcs at supersonic conditions as shown in Fig. 5a. The exact solution is given by

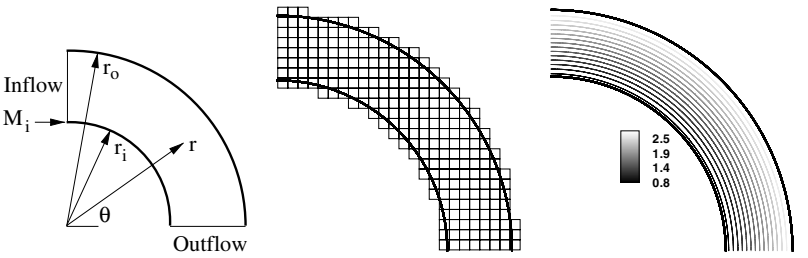
$$\rho = \rho_i \left\{ 1 + \frac{\gamma - 1}{2} M_i^2 \left[1 - \left(\frac{r_i}{r} \right)^2 \right] \right\}^{\frac{1}{\gamma - 1}}$$
$$u = a_i M_i \left(\frac{r_i}{r} \right) \sin\theta, \quad v = -a_i M_i \left(\frac{r_i}{r} \right) \cos\theta, \quad p = \frac{p_i}{\rho_i^\gamma} \rho^\gamma \tag{17}$$

where $a_i = \rho_i = 1$, $p_i = 1/\gamma$, $\gamma = 1.4$, $M_i = 2.25$, $r_i = 1$, and $r_o = 1.382$. As objective function, we use the integral of pressure along the outer arc, which is similar to the common lift and drag boundary integrals

$$\mathcal{J} = \int_0^{\frac{\pi r_o}{2}} p \, \mathrm{d}l = \frac{\pi r_o}{2} p_{r_o} \tag{18}$$

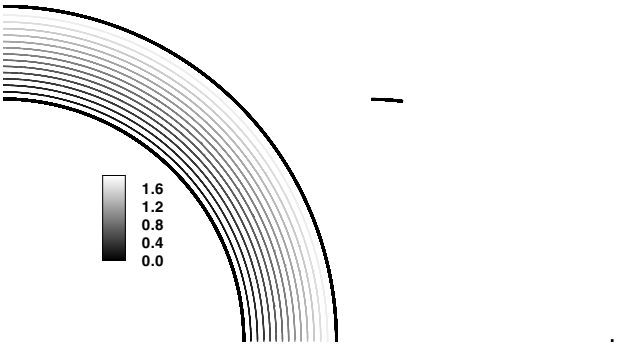
We compute the objective function gradient and the sensitivities of the flow solution with respect to the inlet Mach number, M_i. Straightforward differentiation of Eqs. (17) and (18) with respect to M_i gives the exact solution for the gradient and flow sensitivities. We use the exact solution to specify Dirichlet inlet and outlet boundary conditions for the solution of the flow, flow-sensitivity, and adjoint equations. The problem is solved on a sequence of five nested Cartesian meshes. The coarsest mesh contains 70 cells with a cell-size of 0.1192 for the regular, hexahedral cells. We emphasize that although the selected design variable does not alter the shape of the boundary (terms A and B in Eq. (13) are zero), the boundary discretization changes non-smoothly during mesh refinement.

Before presenting error convergence results, we briefly consider the flow-sensitivity and adjoint solutions shown in Fig. 6 to provide insight into their behavior and physical interpretation. In Fig. 6a, the gradient



(a) Problem setup (b) Example mesh (c) Pressure contours

Fig. 5. Supersonic vortex model problem ($M_i = 2.25$, $r_i = 1$, and $r_o = 1.382$).



(a) $\partial\rho/\partial M_i$ contours

of density with respect to the inlet Mach number, $\partial\rho/\partial M_i$, varies directly with radius and is similar to the pressure field shown in Fig. 5c. The gradient vanishes on the inner arc, because density is independent of Mach number on this boundary. Fig. 6b shows the adjoint solution for the continuity equation. The non-linear variations of the adjoint field correspond to the propagation of point-source mass perturbations, which include interactions with the inner boundary, for an objective function defined by Eq. (18) along the outer arc. This is clearly seen at the outflow boundary of the duct where the adjoint variable vanishes due to the supersonic nature of the flow. Moreover, the adjoint variable is zero in a triangular region upstream of the outflow boundary. This shows that perturbations originating past a certain location on the inner arc, which is a function of the local Mach angle, cannot influence an objective function that is defined only on the outer arc. Lastly, an important observation regarding Fig. 6 is the smooth behavior of both the flow-sensitivity and adjoint solutions in the cut-cell boundary, with no visible irregularities due to the cut-cells.

Fig. 7 summarizes the results of the error convergence study. Fig. 7a shows the error convergence rates in the L_1 norm of density and its gradient with respect to the inlet Mach number. The error convergence rate of the objective function and its gradient is shown in Fig. 7b. The adjoint and flow-sensitivity methods compute identical gradients to round-off error and we display the results obtained via flow sensitivities. The asymptotic

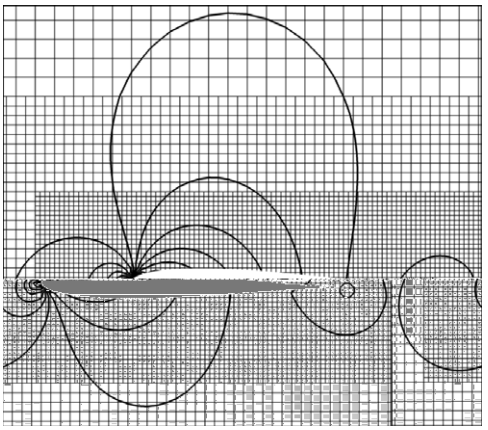


Fig. 8. Near-body mesh and pressure contours for the Joukowski airfoil at $M_\infty = 0.3$ and $\alpha = 1°$. Airfoil is rotated about its trailing edge.

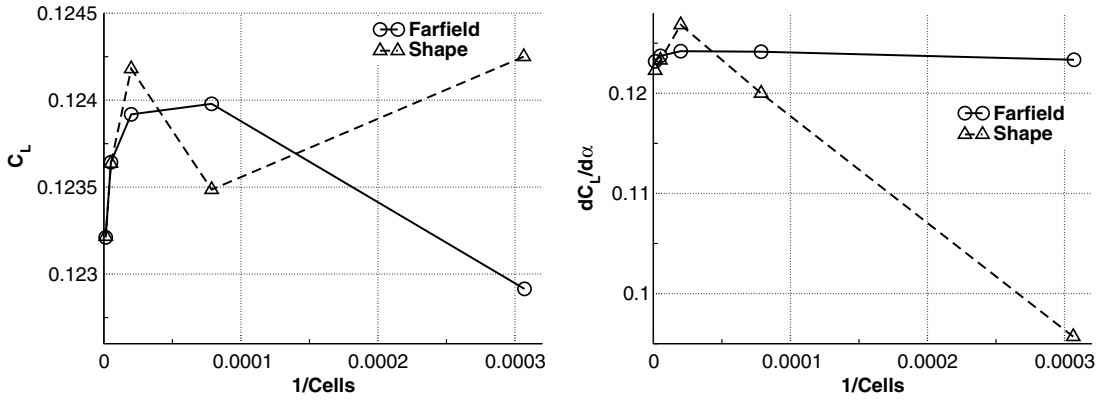
convergence rate of errors, which is measured over the three finest meshes, is just slightly over two. These results are consistent with the second-order spatial discretization of the Euler equations and objective function, thereby verifying the accuracy of the linearization and the convergence of these methods to the continuous problem. Furthermore, the convergence of the objective function and its gradient is not adversely affected by the irregular cells of the domain.

5.2. Shape sensitivities for the Joukowski airfoil

The purpose of this verification test is to study the convergence of gradients for design variables that alter the shape of the boundary. We consider a smooth subsonic flow over the Joukowski airfoil[2] and we choose lift as the objective function. The freestream Mach number is 0.3 and the angle of attack is 1°. We investigate the sensitivity of lift to the angle of attack using two equivalent approaches. First, we consider the influence of the angle of attack via a change in the farfield boundary conditions. This approach is similar to the supersonic vortex problem, because freestream perturbations, such as the Mach number and the angle of attack, do not alter the relationship between the mesh and the airfoil. We contrast this with an angle of attack change implemented via a rigid-body rotation of the airfoil about its trailing edge within a fixed mesh. The mesh-refinement study is performed on a sequence of five nested Cartesian meshes consisting of 3258, 12,718, 50,141, 193,919, and 763,027 cells for each airfoil orientation. The distance to the farfield boundary is 25 chords. The near-body mesh and the pressure contours of the flow solution are shown in Fig. 8 for the second-coarsest mesh of the refinement study.

The results are summarized in Fig. 9. We observe that the objective function and its gradient are converging to essentially the same value. This is in agreement with incompressible flow theory for thin airfoils, which predicts a linear variation of lift with the angle of attack. Moreover, note that the differences between the farfield and rotated airfoil cases for both the objective function and its gradient are decreasing as the mesh is refined. These differences are plotted in Fig. 10 with respect to a normalized cell-size, h. Regression analysis of the data in Fig. 10 reveals that the rate of convergence for the differences in lift is 2.3, which is consistent with second-order spatial discretization. The differences in gradient values, however, vanish at a slower rate. The convergence rate for the gradient is 1.3 – roughly first-order. This reduction in convergence rate is a direct consequence of the noise observed in the computation of the objective function. Unlike the supersonic vortex problem, a perturbation of the angle of attack via airfoil rotation modifies the cut-cell boundary and introduces an error in the objective function proportional to second-order spatial discretization. Nevertheless, the agreement between the farfield and rotated airfoil cases is good over the range of the mesh-refinement

[2] The airfoil is defined using conformal mapping $z = Z + 1/Z$ in the complex Z-plane for a circle centered at $(-0.08, 0)$ with radius of 1.08. The airfoil chord is normalized to unity.



(a) Lift

(b) Gradient of lift with respect to angle of attack

Fig. 9. Convergence study of lift and its gradient due to a change in the angle of attack: implemented via the farfield boundary ("Farfield") and airfoil rotation ("Shape").

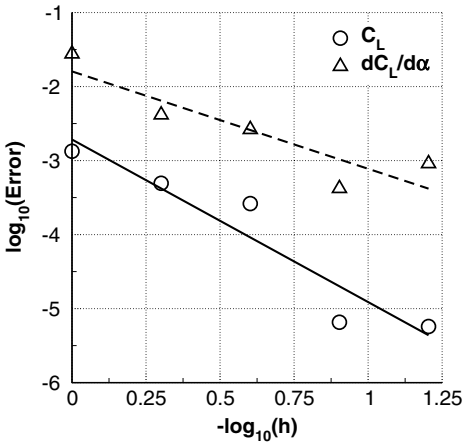


Fig. 10. Convergence of errors for lift and its gradient. Error is measured as the difference between the farfield and rotated airfoil cases. Slope of linear regression lines is −2.2 for lift and −1.3 for its gradient.

study. For example, even on the second-coarsest mesh with roughly 12,000 cells, the gradients are within 2% of the fine-mesh values. Consequently, we expect such gradients to provide reliable descent directions for an optimization algorithm. In the next section, we investigate this hypothesis for several shape-optimization problems.

6. Optimization examples

We present two design examples to demonstrate the effectiveness of the proposed method for aerodynamic shape optimization problems. The optimization problems are solved using the Broyden–Fletcher–Goldfarb–Shanno (BFGS) quasi-Newton method in conjunction with a backtracking line-search [11]. At each step of the line-search, the objective function value and gradient are required to construct a local cubic interpolant. We use the adjoint method for all gradient computations. All computations are performed on Intel 1.6 GHz IA-64 Itanium 2 processors.

6.1. Inverse design

Two inverse design problems are considered for an isolated wing in subsonic and transonic flow. We specify an attainable target pressure distribution based on a known geometry and investigate the convergence characteristics of an optimization that drives an arbitrary input geometry to this target. The effect of limiters on the optimization is examined.

The first example considers subsonic flow at a freestream Mach number of 0.56 and an angle of attack of 3.06°. The subsonic flow does not require the use of limiters ($\phi = 1$ in Eq. (6)), which provides an exact linearization of the flow equations. The flow is smooth everywhere, except for a small region at the sharp trailing edge. The target pressure distribution corresponds to a rectangular wing with an airfoil shape that is based on the ONERA M6 airfoil section [38]. A cubic B-spline curve is used to parameterize the airfoil shape, as shown in Fig. 11. This example allows only vertical displacements of the B-spline control points. Note the symmetric coupling of the two control points at the leading edge in Fig. 11, resulting in an optimization problem with nine design variables. The wing planform (aspect ratio of 3.8) is held fixed during the optimization. The shape sensitivities of the surface triangulation, i.e. the term $\partial\vec{T}/\partial X$ in Eq. (13), are obtained via an analytic linearization of the wing geometry constructors, which include the B-spline curve, linear lofting, and the wing-tip geometry.

The objective function is formulated as a least-squares problem

$$\mathcal{J} = \frac{1}{2}\sum_{i=1}^{nv}(C_p - C_p^*)_i^2 \tag{19}$$

where the target pressure coefficient, C_p^*, is specified at each vertex of the wing triangulation and nv represents the total number of vertices. The initial pressure distribution is obtained by positioning the design variables to approximate the RAE 2822 airfoil section.

Convergence of the flow and adjoint equations is achieved using 32 processors and a 4-level W-cycle multigrid with one pre- and one post-smoothing pass. The computational mesh contains roughly 900,000 cells. Fig. 12 shows typical converge histories of the flow and adjoint solvers in terms of multigrid cycles for density residuals. Full-multigrid startup is used, which corresponds to roughly the first 40 multigrid cycles in Fig. 12. The asymptotic convergence rate of the two solvers is similar. The small difference in the asymptotic slopes is due to omitting an update of the flow-solution gradient prior to residual restriction, which is done to reduce overall wall-clock time [31]. Note that the asymptotic slopes are reached in just 100 multigrid cycles. In terms of the wall-clock time, the convergence of the flow and adjoint solvers is also similar and requires roughly 4 min for each solver.

The accuracy of the computed gradients is presented in Table 1 for the upper surface design variables (see Fig. 11). We compare the adjoint gradient values with centered-difference approximations for the initial wing

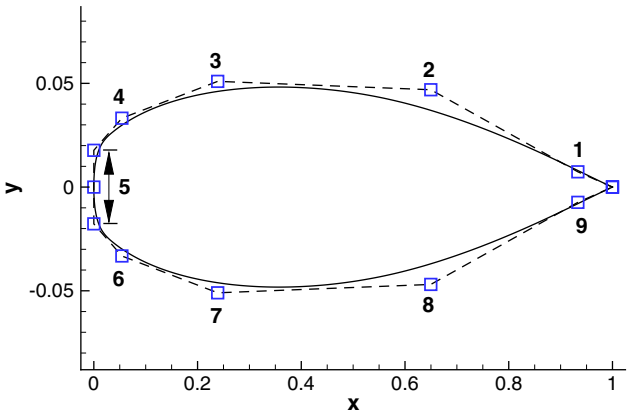


Fig. 11. B-spline control points (symbols) and design variables (labels) for ONERA M6 target airfoil shape.

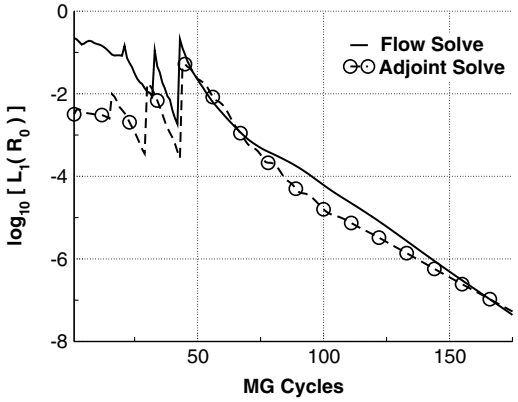


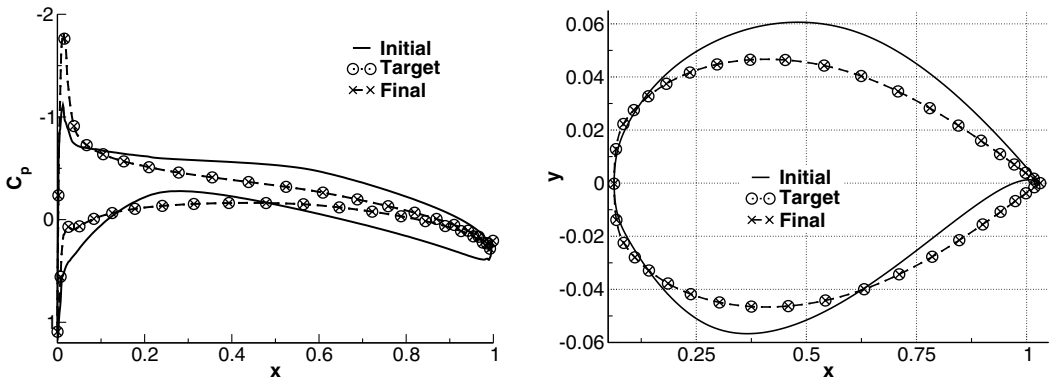
Fig. 12. Example convergence histories for the flow and adjoint solvers using a 4-level W-cycle multigrid (MG), subsonic inverse design $M_\infty = 0.56$ and $\alpha = 3.06^\circ$.

Table 1
Gradient accuracy for subsonic inverse design (upper surface design variables)

Design variable	Finite-difference ($\times 10^4$)	Adjoint (% difference)
1	2.9187	1.7
2	1.6381	1.4
3	3.0351	−0.3
4	2.0301	−3.9
5	−7.1752	0.06

geometry. The agreement in Table 1 is good. We attribute the small differences to changes in the volume mesh discretization due to the finite-difference stepsize. The appearance and disappearance of cut-cells is likely for any finite stepsize. This issue could be avoided by the use of the complex-step method [25]; however, we have not implemented this method due to efficiency reasons and the requirement for access to all source code, which is problematic in a CAD-based design environment.

The optimization results are summarized in Fig. 13. The initial, target, and final pressure distributions near the root region of the wing are shown in Fig. 13a. The corresponding airfoil sections are shown in Fig. 13b. The optimization converges to the target wing shape, and the final and target pressure distributions match to nearly double-precision machine accuracy. This is confirmed in Fig. 14, which summarizes the convergence



(a) Root section pressure distributions (b) Airfoil sections

Fig. 13. Subsonic inverse design problem ($M_\infty = 0.56$ and $\alpha = 3.06^\circ$).

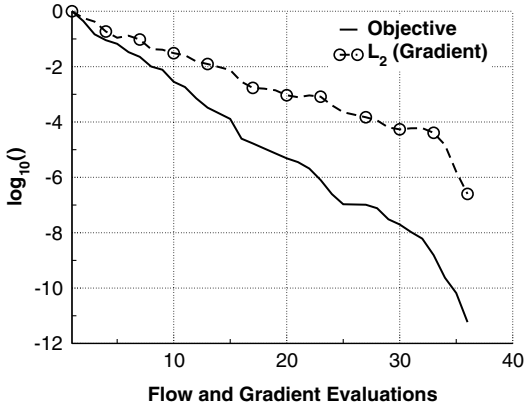


Fig. 14. Optimization convergence history (subsonic inverse design problem).

history of the objective function and gradient. The L_2 norm of the gradient is reduced to single-precision machine accuracy in 36 flow and gradient evaluations. Note the rapid convergence near the optimal solution. This indicates that the BFGS algorithm obtained an accurate estimate of the Hessian matrix using the computed objective function and gradient values.

Next, we consider a more challenging problem of a swept-tapered wing in transonic flow. The target pressure distribution corresponds to the ONERA M6 wing at a freestream Mach number of 0.84 and an angle of attack of 3.06°. The design variables are the same as in the subsonic case, see Fig. 11, and the wing planform (aspect ratio of 3.8, leading-edge sweep angle of 30° and taper ratio of 0.562 with a sharp trailing edge) is held fixed during the optimization. The target wing shape along with the corresponding pressure distribution is shown in Fig. 15. Also shown is the computational mesh, which contains approximately the same number of cells as the subsonic case. Note the strong shock on the upper surface of the wing. We use the van Albada limiter and recall that the limiter is held constant in the linearization.

Fig. 16 shows the initial, target, and final pressure distributions near the root and tip regions of the wing. The corresponding airfoil sections are identical (to plotting accuracy) to the subsonic case in Fig. 13b. The optimization recovers the target wing shape, and the final and target pressure distributions match well even in the sensitive shock and trailing-edge regions. The optimization convergence history is shown in Fig. 17. Note that plotting accuracy for the pressure distribution is obtained in just ten flow and gradient evaluations. The L_2 norm of the gradient is reduced by three orders of magnitude in 24 flow and gradient evaluations, but stalls on the subsequent design iteration. We attribute the stall to the limiter function, which affects the level of convergence of the flow solution and introduces an error in the linearization. While the level of convergence for the optimization is not as deep as in the smooth subsonic example, the optimization is well converged for engineering purposes.

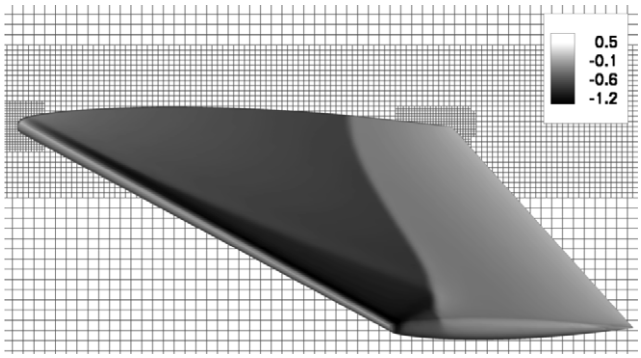
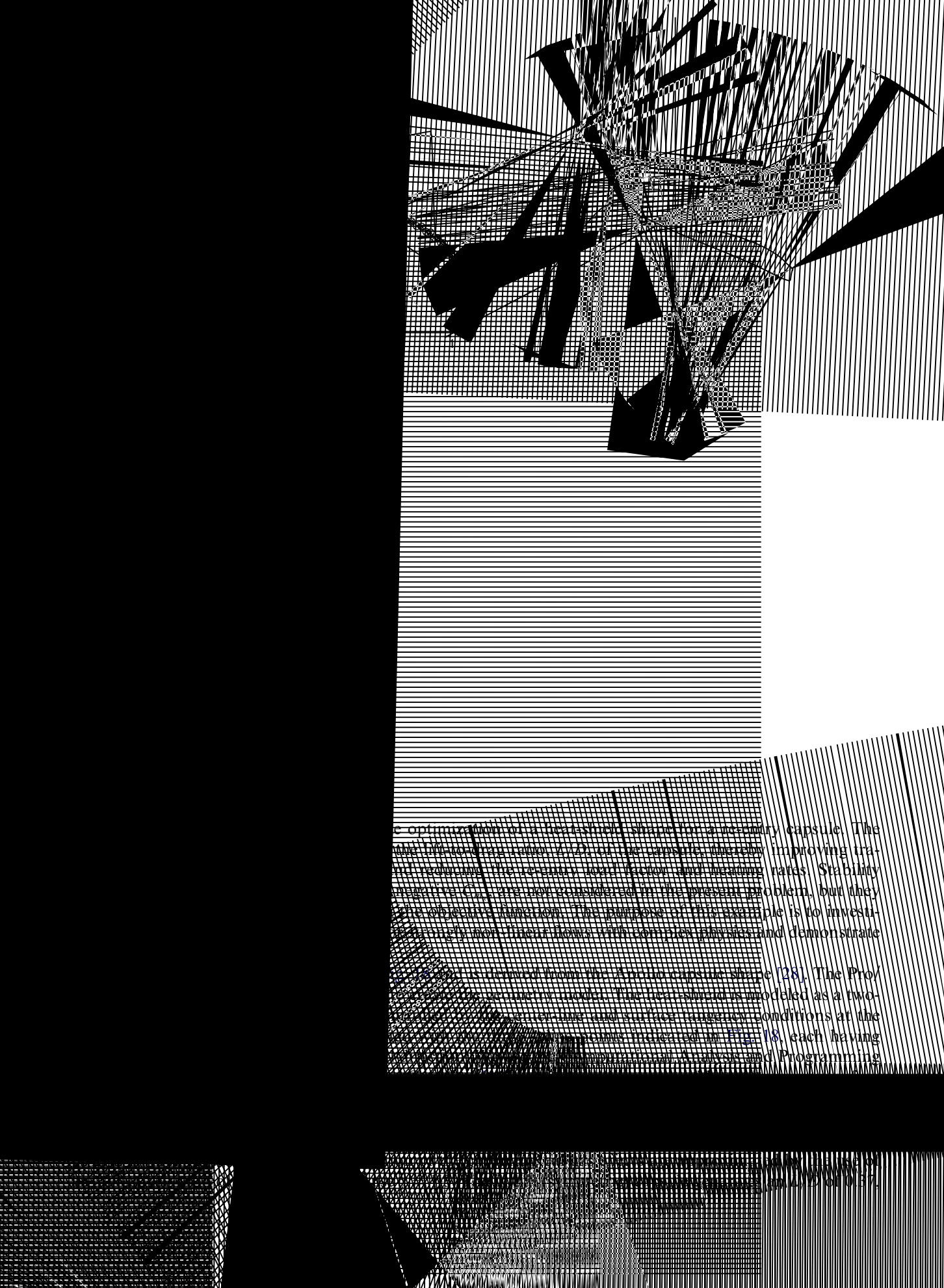


Fig. 15. Near-body mesh (symmetry plane) and wing shape with target C_p contours ($M_\infty = 0.84$ and $\alpha = 3.06°$).



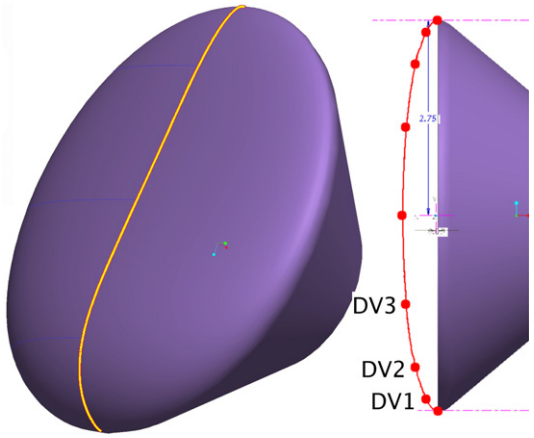


Fig. 18. Parametric-CAD model of capsule shape with design variables (DV) on heat-shield centerline.

which is attained using a center-of-gravity offset given by $x/d = 0.124$, $y/d = -0.03698$, and $z/d = 0$, where d represents the capsule diameter. The target value of L/D is set to 0.4. The objective function is given by

$$\mathcal{J} = \left(\frac{L}{D} - 0.4\right)^2 \tag{20}$$

The volume mesh for the optimization contains roughly 665,000 cells. For the solution of the flow and adjoint equations, we use 64 processors, a 4-level mesh-sequencing startup, and a CFL number of 1.2. The spatial discretization uses a more diffusive *minmod* limiter than the previous transonic example and the agglomerated-normal wall boundary conditions.

Pressure contours of the flow solution for the initial capsule shape are shown in Fig. 19. The main flow features include a strong bow shock and an unsteady wake behind the capsule. Fluctuations in such an energetic flowfield caused some stability problems for the adjoint solver. However, reducing the spatial discretization to first-order in the cut-cells suppressed this behavior and did not significantly impact the values of the aerodynamic coefficients. With these settings, we obtain a residual reduction of approximately three orders of magnitude for the flow and adjoint equations at each design iteration. One factor that we did not investigate in this work is the effect of neglecting the linearization of the limiter on the eigenvalues of the flow-Jacobian matrix, and consequently the convergence of the adjoint time-marching scheme. Similar issues in high Mach

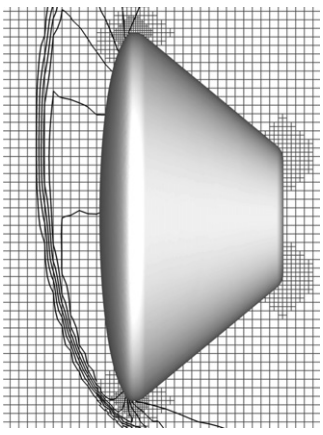


Fig. 19. Near-body mesh and pressure contours on symmetry plane for initial capsule shape ($M_\infty = 10$, $\alpha = 23.5°$, $\gamma = 1.3$).

number flow have been confronted in [34], and this is an area of ongoing work as small scale unsteadiness is common in engineering problems [6].

Gradient accuracy for the initial capsule shape is presented in Table 2. The agreement between the centered-difference and adjoint gradients is not as good as for the subsonic inverse design problem. We attribute the differences to the treatment of the limiter and a greater sensitivity of the finite-difference approximation to the selected stepsize. Nevertheless, the adjoint gradient values provide good descent directions for the optimization. This is shown in Fig. 20, which summarizes the optimization convergence history. The target L/D is reached within five design iterations. The L_2 norm of the gradient is reduced by nearly four orders of magnitude. The initial and final heat-shield shapes are shown in Fig. 21, where the shape modifications are relatively minor yet the improvement in L/D is roughly 8%. Larger gains in L/D can be achieved, but additional constraints are required to avoid shapes with concave regions. The wall-clock time per design iteration is approx-

Table 2
Gradient accuracy for capsule L/D enhancement problem

Design variable	Finite-difference ($\times 10^{-2}$)	Adjoint (%difference)
1	−1.090	16.1
2	−1.125	13.8
3	−0.566	−4.9

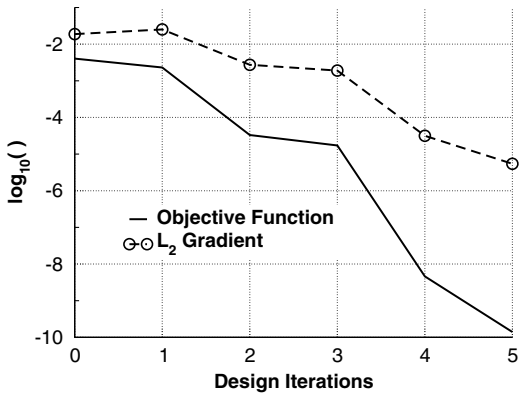


Fig. 20. Capsule L/D optimization convergence history.

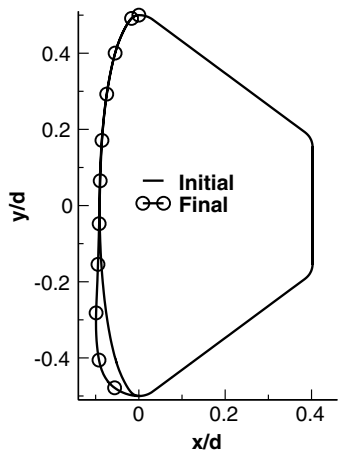


Fig. 21. Initial and final capsule shapes (symmetry plane) for optimization of L/D.

imately 11 min. This time includes the regeneration and triangulation of the CAD part, as well as the flow solution and adjoint gradient computation.

7. Conclusions

A discrete approach for the computation of objective function gradients on embedded-boundary Cartesian meshes has been presented. The effect of shape perturbations in the governing equations was included via a direct linearization of the cut-cell geometric constructors within the mesh generator. In contrast to mesh-perturbation schemes for body-fitted meshes, this approach involves only the near-body cells and does not require explicit connectivity between the surface triangulation and the volume mesh throughout the optimization. The verification studies show that for design variables that do not alter the boundary shape, the error in the gradient value decreases at a rate of $\mathcal{O}(h^2)$. For shape design variables, the convergence rate is reduced to $\mathcal{O}(h)$, which is a consequence of confining the mesh-perturbations to the cut-cells. In all design examples considered, the new approach provides a reliable approximation of the gradient. Furthermore, we obtain a good agreement between the adjoint gradients and finite-difference approximations, with some error caused by treating the limiter as a constant in the linearization. The design examples demonstrate that this simplification does not significantly affect the final designs. The optimizations converge to the performance targets and the L_2 norm of the gradient is reduced by several orders of magnitude. The new approach is well-suited for conceptual design studies and trade-off evaluations where short cycle times are important, and complex geometry with many design variables is common.

Future work should focus on extending this approach to turbulent viscous flow problems and to formulations that involve a general equation of state for real-gas simulations. This would require the linearization of a less diffusive numerical flux function, such as the HLLC scheme [40] or the Hänel et al. [17] modification of the van Leer flux, and viscous effects could be modeled by the use of an interacting boundary-layer approach. An important remaining issue is the robustness of the adjoint and flow-sensitivity solvers for problems with small regions of unsteady flow. It would also be interesting to combine the present approach with adjoint-based error estimation and mesh-adaptation methods to control the accuracy of the objective function throughout the optimization process.

Acknowledgments

The authors gratefully acknowledge Robert Haimes (MIT) for his assistance with CAPRI. The authors would also like to thank Scott Murman (NASA Ames) and Marsha Berger (NYU) for helpful discussions. This work was supported by the NASA Ames Research Center Contract NNA05BF35C.

References

[1] M.J. Aftosmis, M.J. Berger, G. Adomavicius, A parallel multilevel method for adaptively refined Cartesian grids with embedded boundaries, AIAA Paper 2000-0808, Reno, NV, 2000.
[2] M.J. Aftosmis, M.J. Berger, J.E. Melton, Robust and efficient Cartesian mesh generation for component-based geometry, AIAA Journal 36 (6) (1998) 952–960.
[3] W.K. Anderson, V. Venkatakrishnan, Aerodynamic design optimization on unstructured grids with a continuous adjoint formulation, Computers and Fluids 28 (1999) 443–480.
[4] O. Baysal, M.E. Eleshaky, Aerodynamic sensitivity analysis methods for the compressible Euler equations, Journal of Fluids Engineering 113 (4) (1991) 681–688.
[5] M.J. Berger, M.J. Aftosmis, S.M. Murman, Analysis of slope limiters on irregular grids, AIAA Paper 2005-0490, Reno, NV, 2005.
[6] M.S. Campobasso, M.B. Giles, Computing linear harmonic unsteady flows in turbomachines with complex iterative solvers, AIAA Paper 2005-4705, Toronto, ON, 2005.
[7] S.E. Cliff, S.D. Thomas, T.J. Baker, A. Jameson, Aerodynamic shape optimization using unstructured grid methods, AIAA Paper 2002-5550, Atlanta, GA, 2002.
[8] W.J. Coirier, K.G. Powell, An accuracy assessment of Cartesian-mesh approaches for the Euler equations, Journal of Computational Physics 117 (1) (1995) 121–131.
[9] P. Colella, D.T. Graves, B.J. Keen, D. Modiano, A Cartesian grid embedded boundary method for hyperbolic conservation laws, Journal of Computational Physics 211 (1) (2006) 347–366.

[10] A. Dadone, B. Grossman, Efficient fluid dynamic design optimization using Cartesian grids, AIAA Paper 2003-3959, Orlando, FL, 2003.
[11] J.E. Dennis Jr., R.B. Schnabel, Numerical Methods for Unconstrained Optimization and Nonlinear Equations, Prentice-Hall, Englewood Cliffs, NJ, 1983.
[12] R. Duvigneau, D. Pelletier, Evaluation of nearby flows by a shape sensitivity equation method, AIAA Paper 2005-127, 2005.
[13] R.P. Dwight, Robust mesh deformation using the linear elasticity equations, in: Proceedings of the Fourth International Conference on Computational Fluid Dynamics (ICCFD 4), Ghent, Belgium, 2006.
[14] M.B. Giles, On the Use of Runge–Kutta Time-Marching and Multigrid for the Solution of Steady Adjoint Equations, Oxford University Computing Laboratory 00/10, June 2000.
[15] M.B. Giles, N.A. Pierce, An introduction to the adjoint approach to design, Flow Turbulence and Combustion 65 (3/4) (2000) 393–415.
[16] R. Haimes, C. Crawford, Unified geometry access for analysis and design, Technical Report, 12th International Meshing Roundtable, Santa Fe, NM, 2003.
[17] D. Hänel, R. Schwane, G. Seider, On the accuracy of upwind schemes for the solution of the Navier–Stokes equations, AIAA Paper 87-1105, 1987.
[18] W.P. Huffman, R.G. Melvin, D.P. Young, F.T. Johnson, J.E. Bussoletti, M.B. Bieterman, C.L. Hilmes, Practical design and optimization in computational fluid dynamics, AIAA Paper 93-3111, 1993.
[19] A. Jameson, Aerodynamic design via control theory, Journal of Scientific Computing 3 (1988) 233–260.
[20] A. Jameson, Aerodynamic shape optimization using the adjoint method, Lecture Notes, von Karman Institute for Fluid Dynamics, Brussels, Belgium, 2003.
[21] A. Jameson, S. Kim, Reduction of the adjoint gradient formula in the continuous limit, AIAA Paper 2003-0040, 2003.
[22] A.J. Keane, Wing optimization using design of experiment, response surface, and data fusion methods, Journal of Aircraft 40 (4) (2003) 741–750.
[23] C.S. Kim, C. Kim, O.H. Rho, Sensitivity analysis for the Navier–Stokes equations with two-equation turbulence models, AIAA Journal 39 (5) (2001) 838–845.
[24] J. Lu, An a posteriori error control framework for adaptive precision optimization using discontinuous Galerkin finite element method, Ph.D. thesis, Massachusetts Institute of Technology, 2005.
[25] J.R.R.A. Martins, P. Sturdza, J.J. Alonso, The complex-step derivative approximation, ACM Transactions on Mathematical Software 29 (3) (2003) 245–262.
[26] D.J. Mavriplis, Formulation and multigrid solution of the discrete adjoint for optimization problems on unstructured meshes, AIAA Journal 44 (1) (2006) 42–50.
[27] B. Mohammadi, O. Pironneau, Applied Shape Optimization for Fluids, Oxford University Press, New York, USA, 2001.
[28] W.C. Moseley Jr., R.E. Graham, J.E. Hughes, Aerodynamic stability characteristics of the Apollo command module, NASA TN D-4688, 1968.
[29] A. Nelson, M. Nemec, M.J. Aftosmis, T.H. Pulliam, Aerodynamic optimization of rocket control surfaces using Cartesian methods and CAD geometry, AIAA Paper 2005-4836, Toronto, ON, 2005.
[30] M. Nemec, M.J. Aftosmis, Adjoint algorithm for CAD-based shape optimization using a Cartesian method, AIAA Paper 2005-4987, Toronto, ON, 2005.
[31] M. Nemec, M.J. Aftosmis, S.M. Murman, T.H. Pulliam, Adjoint formulation for an embedded-boundary Cartesian method, AIAA Paper 2005-0877, Reno, NV, 2005.
[32] M. Nemec, D.W. Zingg, Newton–Krylov algorithm for aerodynamic design using the Navier–Stokes equations, AIAA Journal 40 (6) (2002) 1146–1154.
[33] E.J. Nielsen, W.K. Anderson, Recent improvements in aerodynamic design optimization on unstructured meshes, AIAA Journal 40 (6) (2002) 1155–1163.
[34] E.J. Nielsen, B. Kleb, Efficient construction of discrete adjoint operators on unstructured grids by using complex variables, AIAA Paper 2005-0324, Reno, NV, 2005.
[35] E.J. Nielsen, M.A. Park, Using an adjoint approach to eliminate mesh sensitivities in computational design, AIAA Paper 2005-0491, Reno, NV, 2005.
[36] J.J. Reuther, Aerodynamic shape optimization using control theory, Ph.D. thesis, University of California Davis, 1996.
[37] D.L. Rodriguez, Propulsion/Airframe integration and optimization on a supersonic business jet, AIAA Paper 2007-1048, 2007.
[38] V. Schmitt, F. Charpin, Pressure distributions on the ONERA-M6-wing at transonic Mach numbers, in: Experimental Data Base for Computer Program Assessment, AGARD-R-138, 1979.
[39] O. Soto, R. Löhner, On the computation of flow sensitivities from boundary integrals, AIAA Paper 2004-0112, 2004.
[40] E.F. Toro, M. Spruce, W. Speares, Restoration of the contact surface in the HLL-Riemann solver, Shock Waves 4 (1) (1994) 25–34.
[41] A.H. Truong, C.A. Oldfield, D.W. Zingg, Mesh movement for a discrete-adjoint Newton–Krylov algorithm for aerodynamic optimization, AIAA Paper 2007-3952, Miami, FL, 2007.
[42] B. van Leer, Flux–vector splitting for the Euler equations, ICASE Report 82-30, 1982.
[43] Z. Yang, D.J. Mavriplis, Unstructured dynamic meshes with higher-order time integration schemes for the unsteady Navier–Stokes equations, AIAA Paper 2005-1222, Reno, NV, 2005.
[44] D.P. Young, R.G. Melvin, M.B. Bieterman, F.T. Johnson, S.S. Samant, A locally refined rectangular grid finite element method: application to computational fluid dynamics and computational physics, Journal of Computational Physics 92 (1) (1991) 1–66.