AIAA-2001-0596

Recent Improvements in Aerodynamic Design Optimization On Unstructured Meshes

Eric J. Nielsen* and W. Kyle Anderson†
NASA Langley Research Center, Hampton, VA 23681-2199

Abstract

Recent improvements in an unstructured-grid method for large-scale aerodynamic design are presented. Previous work had shown such computations to be prohibitively long in a sequential processing environment. Also, robust adjoint solutions and mesh movement procedures were difficult to realize, particularly for viscous flows. To overcome these limiting factors, a set of design codes based on a discrete adjoint method is extended to a multiprocessor environment using a shared memory approach. A nearly linear speedup is demonstrated, and the consistency of the linearizations is shown to remain valid. The full linearization of the residual is used to precondition the adjoint system, and a significantly improved convergence rate is obtained. A new mesh movement algorithm is implemented and several advantages over an existing technique are presented. Several design cases are shown for turbulent flows in two and three dimensions.

Introduction

With the advent of modern computer architectures, aerodynamic designers have sought to make use of high-fidelity computational fluid dynamics (CFD) codes in their everyday design efforts. While considerable progress has been made towards this goal, realistic use of such tools remains hindered by the extreme computational burden associated with such an endeavor.

A large focus has recently been placed on design algorithms. In the area of gradient-based optimization, research has focused on several methods for obtaining sensitivity information, and many of these approaches rely on an adjoint-variable formulation for efficiently computing sensitivity derivatives. The adjoint technique is particularly attractive for aerodynamic design problems in which there are a large number of design variables, yet relatively few constraints. Examples of both continuous and discrete approaches to this method can be found in Refs. 1-10.

In Refs. 1-4, a discrete adjoint technique has been implemented on unstructured grids for two- and three-dimensional flows. This work was primarily aimed at performing accurate linearizations of Reynolds-averaged Navier-Stokes solvers, using both compressible and incompressible formulations. Results indicated highly accurate sensitivity information for fully turbulent flows. However, the cost of such computations in a sequential-processing environment prevented large-scale design cases from being pursued. The preconditioning strategy used for the adjoint system in these references was based on a first-order linearization of the residual and often led to poor convergence rates. In addition, experience showed that the combination of a distance function approach and tension-spring analogy used for mesh movement was insufficient when large changes in the geometry were necessary. This procedure was also intolerant of initial meshes with poor quality.

In the current work, the linearizations developed in Refs. 1-4 are modified to run in a parallel processing environment. The domain decomposition and parallelization strategies are discussed, resulting speedups are demonstrated, and the linearizations are shown to remain consistent. A new preconditioning strategy for the adjoint solver is implemented and significantly improved convergence is demonstrated for turbulent flow. A new mesh movement strategy based on modified linear elasticity theory is also adopted, and several advantages over the previous approach are presented. Several design cases are also shown.

Nomenclature

c_l, c_d	Lift and drag coefficients
c	Chord
$\boldsymbol{D}$	Vector of design variables
f	Cost function
$\boldsymbol{I}$	Identity matrix
L	Lagrangian function
$\boldsymbol{Q}$	Vector of dependent variables
$\boldsymbol{R}$	Discretized residual vector
t	Time
u, v	Nodal displacements
$\vec{\boldsymbol{V}}$	Vector of nodal displacements
V	Volume of control volume
$\boldsymbol{X}$	Computational mesh
Λ	Vector of costate variables
ν	Poisson's ratio

Design Methodology

Flow Equations

The governing flow equations are the Reynolds-averaged Navier-Stokes equations,[11] coupled with the one-equation turbulence model of Spalart and Allmaras.[12] The flow solvers used in the current work are described at length in Refs. 4, 13, and 14. The codes use an implicit, upwind, finite-volume discretization, in which the dependent variables are stored at the mesh vertices. Inviscid fluxes at cell interfaces are computed using the upwind schemes of Roe[15], van Leer[16], or Osher[17]. Viscous fluxes are formed using an approach equivalent to a central-difference Galerkin procedure. Temporal discretization is performed using a backward-Euler time-stepping scheme. The meshes used in this study have been generated using the software described in Refs. 18 and 19.

An approximate solution of the linear system of equations formed at each time step is obtained using several iterations of a point-iterative scheme in which the nodes are updated in an even-odd fashion, resulting in a Gauss-Seidel-type method.

*Research Scientist, Computational Modeling and Simulation Branch, Aerodynamics, Aerothermodynamics, and Acoustics Competency, Member AIAA.

†Formerly Senior Research Scientist, Computational Modeling and Simulation Branch, Aerodynamics, Aerothermodynamics, and Acoustics Competency. Assoc. Fellow AIAA.

The turbulence model is solved separately from the flow equations at each time step, using a backward-Euler time-stepping scheme. The resulting linear system is solved using the same point-iterative scheme employed for the flow equations. The turbulence model is integrated all the way to the wall without the use of wall functions.

Adjoint and Design Equations

Given a steady-state flow solution in the form of $\boldsymbol{R}(\boldsymbol{D}, \boldsymbol{Q}, \boldsymbol{X}) = 0$, a Lagrangian function can be defined as

$$L(\boldsymbol{D}, \boldsymbol{Q}, \boldsymbol{X}, \Lambda) = f(\boldsymbol{D}, \boldsymbol{Q}, \boldsymbol{X}) + \Lambda^T \boldsymbol{R}(\boldsymbol{D}, \boldsymbol{Q}, \boldsymbol{X}) \qquad (1)$$

where $f(\boldsymbol{D}, \boldsymbol{Q}, \boldsymbol{X})$ represents a cost function to be minimized and Λ represents a vector of Lagrange multipliers, or costate variables. Differentiating this expression yields the following:

$$\frac{dL}{d\boldsymbol{D}} = \left\{\frac{\partial f}{\partial \boldsymbol{D}} + \left[\frac{\partial \boldsymbol{X}}{\partial \boldsymbol{D}}\right]^T \frac{\partial f}{\partial \boldsymbol{X}}\right\} + \qquad (2)$$

$$\left[\frac{\partial \boldsymbol{Q}}{\partial \boldsymbol{D}}\right]^T \left\{\frac{\partial f}{\partial \boldsymbol{Q}} + \left[\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{Q}}\right]^T \Lambda\right\} + \left\{\left[\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{D}}\right]^T + \left[\frac{\partial \boldsymbol{X}}{\partial \boldsymbol{D}}\right]^T \left[\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{X}}\right]^T\right\} \Lambda$$

Since the vector of costate variables is essentially arbitrary, the coefficient multiplying $[\partial \boldsymbol{Q}/\partial \boldsymbol{D}]^T$ can be eliminated using the following equation:

$$\left[\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{Q}}\right]^T \Lambda = -\frac{\partial f}{\partial \boldsymbol{Q}} \qquad (3)$$

Eq. 3 represents the discrete adjoint equation for the design problem. Once the solution for Λ has been formed, the remaining terms in Eq. 2 can be evaluated to give the desired sensitivity information:

$$\frac{\partial L}{\partial \boldsymbol{D}} = \left\{\frac{\partial f}{\partial \boldsymbol{D}} + \left[\frac{\partial \boldsymbol{X}}{\partial \boldsymbol{D}}\right]^T \frac{\partial f}{\partial \boldsymbol{X}}\right\} + \left\{\left[\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{D}}\right]^T + \left[\frac{\partial \boldsymbol{X}}{\partial \boldsymbol{D}}\right]^T \left[\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{X}}\right]^T\right\} \Lambda \qquad (4)$$

The adjoint equation given in Eq. 3 represents a linear set of equations for the costate variables Λ. Although this system can be solved directly using GMRES,[20] a time-like derivative is added and the solution is obtained by marching in time, much like the flow solver:

$$\left\{\frac{V}{\Delta t}\boldsymbol{I} + \left[\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{Q}}\right]^T\right\} \Delta^n \Lambda = -\frac{\partial f}{\partial \boldsymbol{Q}} - \left[\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{Q}}\right]^T \Lambda^n \qquad (5)$$

where

$$\Lambda^{n+1} = \Lambda^n + \Delta^n \Lambda \qquad (6)$$

The time term can be used to increase the diagonal dominance for cases in which GMRES alone would tend to stall. This ultimately results in a more robust adjoint solver.

In Refs. 1-5, an incomplete LU decomposition of the matrix obtained from a first-order accurate discretization is used to precondition the linear system. The preconditioning is applied on the left and no fill-in is allowed (ILU[0]).[21]

Domain Decomposition Methodology

In the current work, the mesh partitioner MeTiS[22] is used to divide the original mesh into subdomains suitable for a parallel environment. Given the connectivities associated with each node in the mesh and the number of partitions desired, MeTiS returns an array that designates a partition number for each node in the mesh. The user is then responsible for extracting the data structures required by the specific application.

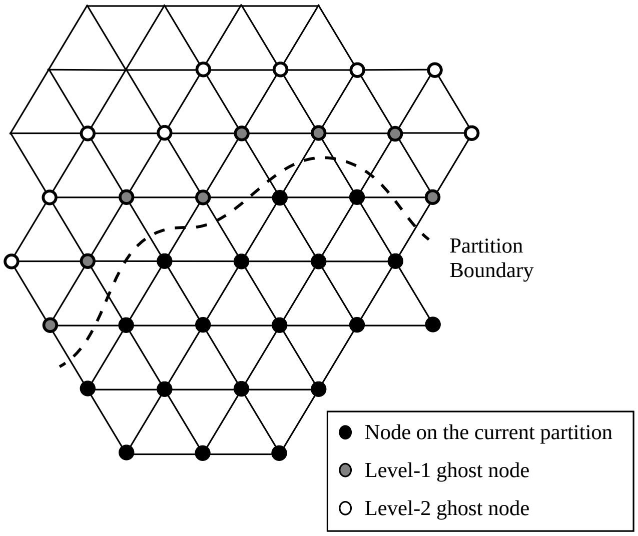

Figure 1. Information required beyond partition boundaries.

Due to the gradient terms used in the reconstruction procedure, achieving second-order accuracy in the flow solver requires information from the neighbors of each mesh point as well as the points adjacent to these neighbors. In the present implementation, the gradients of the dependent variables are first computed on each mesh partition and then the results are scattered onto neighboring partitions. This approach dictates that a single level of “ghost” nodes be stored on each processor. These ghost nodes that are connected to mesh points on the current partition are referred to as “level-1” nodes. Similarly, the neighbors of level-1 nodes that do not lie on the current partition are designated “level-2” nodes. This terminology is illustrated graphically in Fig. 1.

The adjoint solver requires similar information; however, unlike the flow solver, residual contributions must be written into off-processor memory locations associated with level-2 mesh points. This implies that a second level of ghost information must be retained along partition boundaries.

Software has been developed to extract the required information from a pre-existing mesh based on the partitioning array provided by MeTiS. This domain decomposition operation is done prior to performing any computations. The user is also able to read in existing subdomains and their corresponding solution files and repartition as necessary. This capability is useful in the event that additional processors become available or processors currently being employed must be surrendered to other users. In addition, software has been developed that reassembles partition information into global files and aids in post-processing the solutions.

Parallelization Strategy

Each of the codes has been modified to run in a multiprocessor environment using a shared memory implementation. This approach has been chosen because the primary hardware to be utilized is a Silicon Graphics Origin 2000 system. In the current implementation, ghost information is exchanged across partition boundaries by loading data into global shared arrays which are accessible from each processor. Simple compiler directives specific to the Origin 2000 system are used to spawn child processes for each partition in the mesh. This approach scales well and is readily extendable to a message-passing or OpenMP[23] implementation. The convergence rate of the flow solution is independent of the number of processors, whereas the convergence of the adjoint solver varies slightly since the preconditioner is only applied locally on each mesh partition.

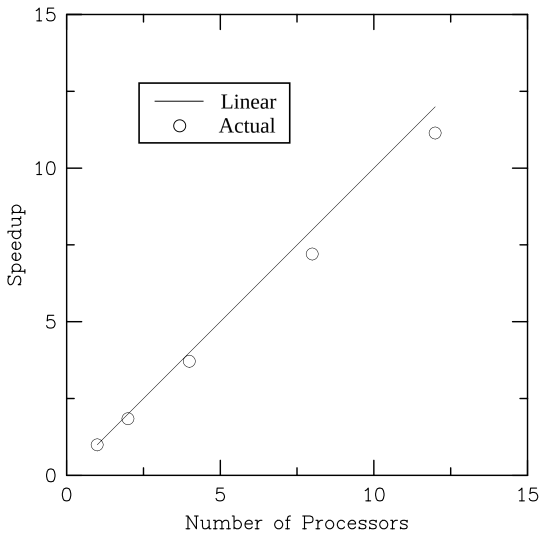

Figure 2. Parallel speedup obtained for the flow solver.

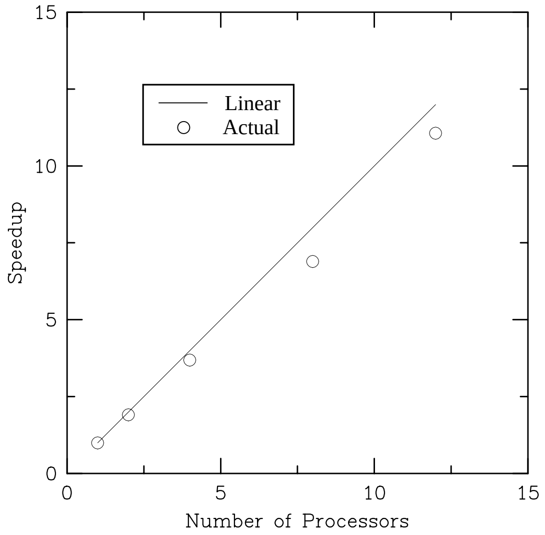

Figure 3. Parallel speedup obtained for the adjoint solver.

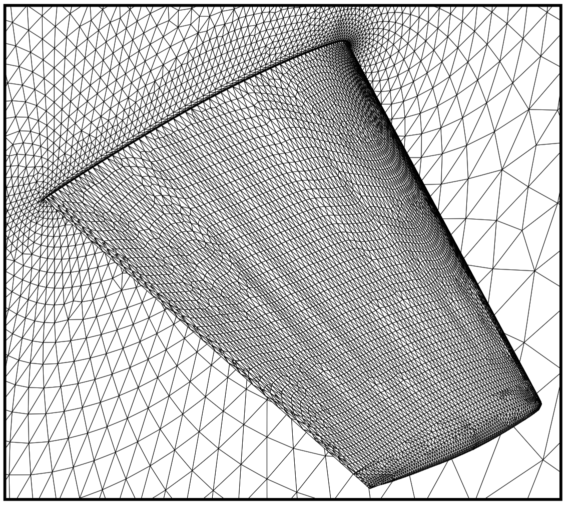

Figure 4. Surface mesh for viscous ONERA M6 wing.

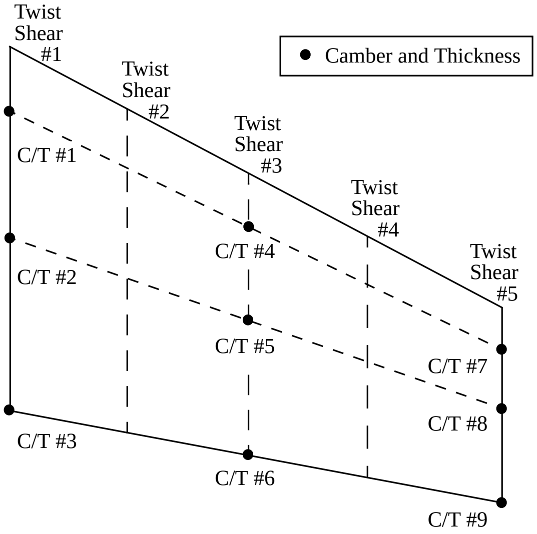

Figure 5. Location of design variables for ONERA M6 wing.

Table 1. Sensitivity derivatives for turbulent flow over ONERA M6 wing computed in parallel.

Design Variable	Finite Difference	Adjoint	Percent Error
Camber #3	2.7762	2.7763	0.004%
Thickness #4	-0.03970	-0.03971	0.025%
Twist #4	0.00747	0.00747	0.000%
Shear #1	0.62023	0.62050	0.044%

The speedup obtained by parallelizing the flow and adjoint solvers is demonstrated in Figs. 2 and 3. It can be seen that a nearly linear speedup is obtained. For this test, turbulent flow over the ONERA M6 wing shown in Fig. 4 is computed. The mesh contains 359,536 nodes with a wall spacing of 2×10^{-6} of the mean aerodynamic chord (MAC). The surface mesh consists of 9,129 nodes.

To verify that the linearizations performed in Refs. 1-4 have remained consistent through the port to the parallel environment, sensitivity derivatives obtained using the parallel solvers on eight processors are compared with centered finite differences. Here, turbulent flow over an ONERA M6 wing is computed using a freestream Mach number of 0.3, an angle of attack of $2°$, and a Reynolds number of $5{\times}10^{6}$ based on the MAC. The mesh used for this case consists of 16,391 nodes. All results have been converged to machine accuracy, and a step size of 1×10^{-5} has been used for the finite-difference computations. For this case, the cost function is a linear combination of lift and drag, and the design variables generated using the software described in Ref. 24 are depicted in Fig. 5. It can be seen from Table 1 that the derivatives are highly consistent.

Adjoint Preconditioning Scheme

In Refs. 1-5, a preconditioned GMRES algorithm has been used to solve Eq. 3. In these references, an incomplete LU-factorization with no fill-in allowed [ILU(0)] is employed as the preconditioner. The factorization is based on the first-order linearization of the residual, thereby avoiding excessive storage penalties associated with the higher-order stencil for the inviscid fluxes. It has been shown in Ref. 4 that the GMRES algorithm may stall and a converged adjoint solution may be difficult to obtain using this preconditioner, particularly for viscous flows. This has been found to be the case for both two- and three-dimensional problems.

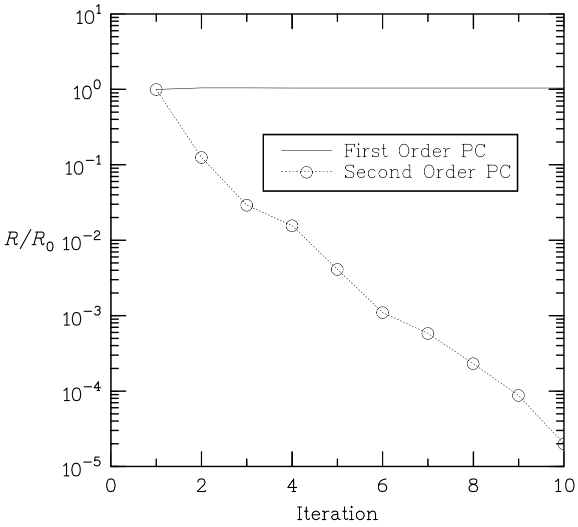

Figure 6. Convergence of the adjoint solution for different preconditioners.

In an effort to develop a more robust adjoint solver, an improved ILU(0) preconditioning technique based on the complete linearization of the residual is employed in the current work. As shown in Ref. 25, the additional memory required for storing the complete linearization is roughly four times that of the first-order matrix for three-dimensional problems. This requirement can be somewhat alleviated by utilizing half-precision storage for these terms. As described in Refs. 3 and 4, the linearizations required for the matrix-vector products in the GMRES algorithm are stored for the nearest-neighbor terms; these linearizations are also stored in half-precision in the current work. Experiments have shown that this strategy yields a total memory requirement of about 50% more than the previous version of the adjoint solver.

To demonstrate the improved performance using the higher-order preconditioner, adjoint solutions are computed in parallel for turbulent flow over the ONERA M6 wing shown in Fig. 4 using eight processors. The freestream Mach number is 0.84, the angle of attack is $3.06°$, and the Reynolds number is 5×10^6 based on the MAC. For this case, 10 GMRES cycles are used with 10 search directions and 5 restarts. Results for the first- and second-order preconditioning strategies are shown in Fig. 6. It can be seen that the solver based on the first-order preconditioner fails to converge the solution, whereas the method employing the complete linearization steadily reduces the residual by nearly five orders of magnitude.

Mesh Movement Strategy

As stated in Refs. 1-5, a combination of a distance function approach and a tension-spring analogy has previously been employed as a means for modifying volume meshes as the geometric shape is changed throughout the design process. It has been found that this algorithm lacks the robustness necessary for the design environment, particularly for large surface deformations, meshes with highly distorted cells, and essentially all three-dimensional geometries. For this reason, a new approach based on modified linear elasticity theory has been implemented.

In the approach taken in the current work, it is assumed that the computational mesh obeys the isotropic linear elasticity relations which take the following form in two dimensions:[26]

$$\nabla^2 u + \frac{1}{1-2\nu}\frac{\partial}{\partial x}\nabla \cdot \vec{V} = 0 \qquad (7)$$

$$\nabla^2 v + \frac{1}{1-2\nu}\frac{\partial}{\partial y}\nabla \cdot \vec{V} = 0 \qquad (8)$$

where ν is Poisson's ratio and the nodal displacement vector is given by $\vec{V} = u\hat{i} + v\hat{j}$. Despite the assumption of an isotropic material, a spatially-varying value of Poisson's ratio is used in order to maintain the physical integrity of highly skewed cells. This value has been chosen based on heuristics and is set so that the coefficient $1/(1-2\nu)$ is equal to the aspect ratio of the local cell. In this manner, low aspect ratio cells mimic compressible materials such as cork, while high aspect ratio cells tend to behave in an incompressible fashion, much like rubber. Since the nodes on the surface are constrained, the high aspect ratio cells in the near-wall region are not susceptible to compression. A similar mesh movement scheme has also been utilized in Ref. 27. Here, anisotropy in Poisson's ratio is achieved by neglecting the Jacobian associated with the transformation between physical and computational coordinates. In this manner, Poisson's ratio is implicitly determined by the cell volumes, so that small cells deform less.

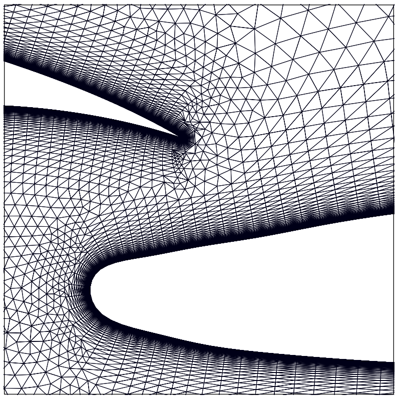

Figure 7. Near-field view of baseline mesh.

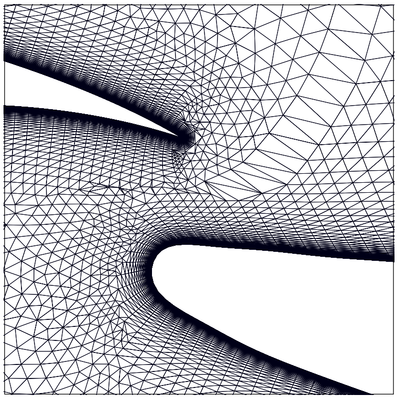

Figure 8. Near-field view of mesh after applying distance function/ tension-spring analogy.

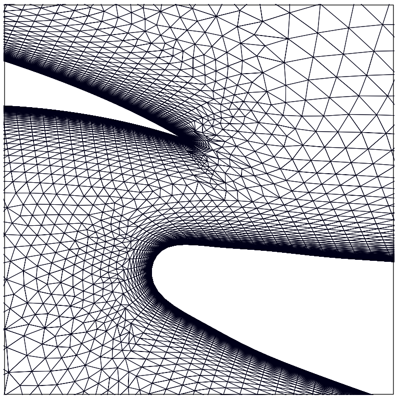

Figure 9. Near-field view of mesh after applying modified linear elasticity method.

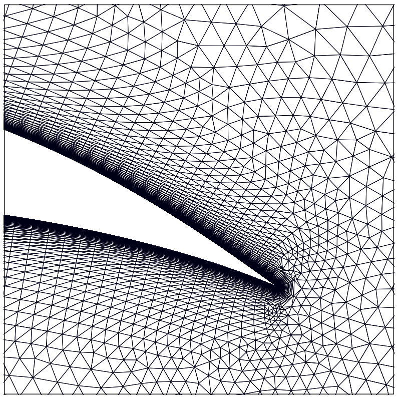

Figure 10. Near-field view of mesh with flap in baseline position.

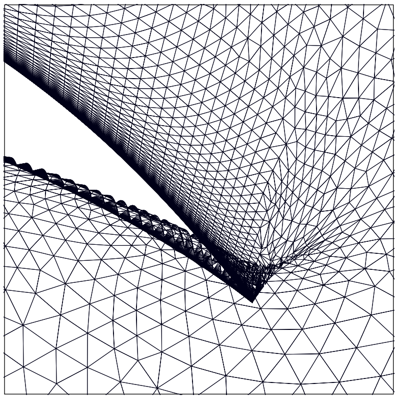

Figure 11. Near-field view of mesh with flap rotated using the distance function/tension-spring analogy.

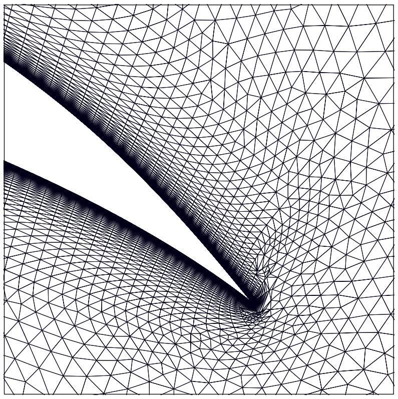

Figure 12. Near-field view of mesh with flap rotated using modified linear elasticity method.

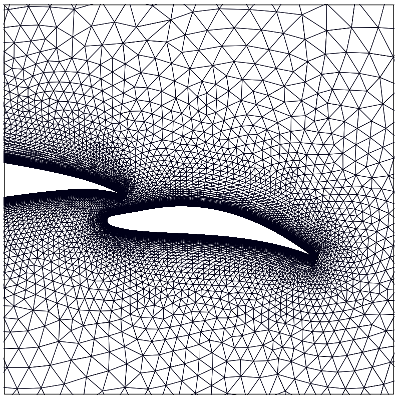

Figure 13. Mesh with flap in baseline position.

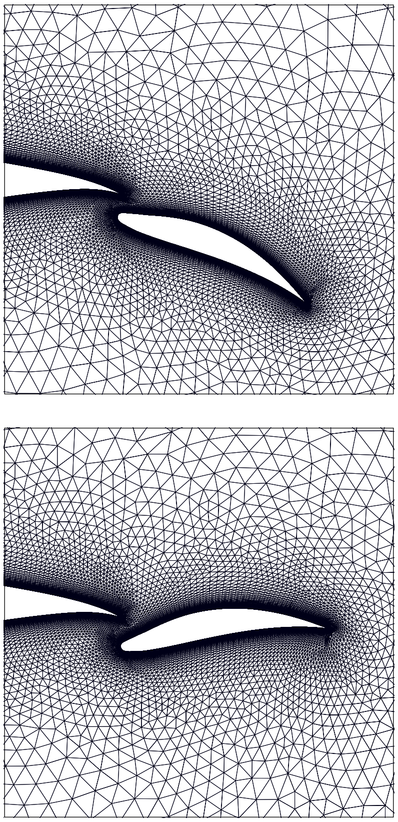

Figure 14. Mesh with flap translated $\Delta x / c = 0.02$ and rotated $\pm 15°$.

To illustrate the advantage over the distance function/tension-spring analogy, the flap on a multielement airfoil has been deflected $15°$ and each of the mesh movement strategies has been applied. Figure 7 shows a near-field view of the baseline mesh in the region between the main element and flap. Figures 8 and 9 show the meshes resulting from the distance function/tension-spring and elasticity methods, respectively. It can be seen that the distance function/tension-spring analogy allows gaps to form in the mesh, whereas the elasticity approach pulls in nearby material to fill the voids.

It has been found that the elasticity approach also allows for significantly larger geometric deformations. In a similar test, the flap of a multielement airfoil has been deflected from its baseline position shown in Fig. 10. As can be seen from Figs. 11 and 12, the distance function/tension-spring approach has yielded an invalid mesh, while the elasticity formulation has handled the deformation in an acceptable manner. Similarly, when a series of flap translations and rotations is applied to the geometry shown in Fig. 13, the meshes resulting from the elasticity technique maintain a high degree of quality as shown in Fig. 14.

To further quantify the differences between the two mesh movement schemes, derivatives of lift and drag due to horizontal translations of a main element and flap are examined. Ideally, the derivative due to a translation of the flap should be equal and opposite in sign to a derivative due to an equal and opposite translation of the main element. In practice however, this relationship is affected by changes in the topology of the mesh due to the manner in which it varies during a shape modification.[5]

To demonstrate this behavior, derivatives of lift and drag due to equal and opposite horizontal translations of the main element and flap on the two-element airfoil shown in Fig. 15 have been computed for a turbulent flow. For this case, the freestream Mach num-

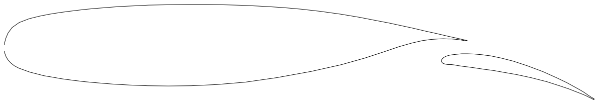

Figure 15. Geometry used for translation derivatives.

Table 2. Derivatives of lift and drag due to flap and main element translation using the distance function/tension-spring analogy.

Derivative	x_{main}	x_{flap}	$\sum x$
$\partial c_l / \partial x$	-1.4785	2.4033	0.9248
$\partial c_d / \partial x$	0.0183	0.0277	0.0460

Table 3. Derivatives of lift and drag due to flap and main element translation using linear elasticity.

Derivative	x_{main}	x_{flap}	$\sum x$
$\partial c_l / \partial x$	-3.8064	3.8671	0.0607
$\partial c_d / \partial x$	0.1722	-0.1615	0.0107

ber is 0.25, the Reynolds number is 9×10^6, and the angle of attack is $5°$.

Table 2 shows the lift and drag derivatives due to translations of the main element and flap using the distance function/tension-spring analogy, and the last column is the sum of these two derivatives which ideally would be zero. However, it can be seen from the table that the derivatives are not at all equal in magnitude, and the drag derivatives are not even of opposite sign. This inconsistency would be expected to have an adverse effect on an optimization procedure. Table 3 shows the same derivatives obtained using the linear elasticity formulation. Although these derivatives do not sum exactly to zero, they do exhibit opposite signs and are much closer in magnitude. This tendency has been observed in several cases and indicates that the linear elasticity formulation maintains the mesh topology in a more consistent fashion.

Design Cases

Recovery of Experimental Flap Configuration

The first test case is a two-dimensional turbulent flow problem for which a target pressure distribution is sought. An experimental study of the multielement airfoil geometry shown in Fig. 16 has been previously performed, and it can be seen from Fig. 17 that computations using the baseline geometry are in disagreement with the experimental results. The model used in the experiment had a non-uniform gap and overlap across the span, and the flap deflected at high dynamic pressures. The goal of the current work is to determine a new position of the flap using the pressure distribution obtained in the experiment. The improved mesh movement capability described above allows for the flap adjustment required by such a problem.

The freestream Mach number is 0.7, the angle of attack is $1.5°$, and the Reynolds number is 30×10^6. For this case, the design variables are the rotation and x- and y-translations of the flap. After 5 design cycles, the flap has been rotated $3.5°$ and repositioned as shown in Fig. 16, although very little change has occurred after the first design cycle. It can be seen from Fig. 17 that the resulting agreement with the experimental results is significantly improved. Although not shown, an angle of attack sweep verifies that the new position of the flap improves the agreement with the experiment across the range of angles of attack.

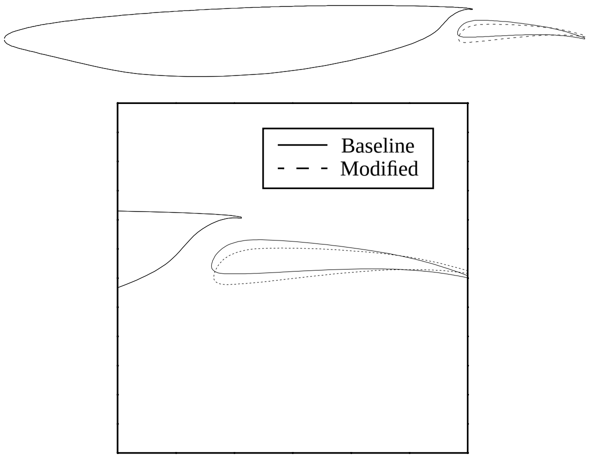

Figure 16. Baseline and modified geometries for multielement airfoil problem.

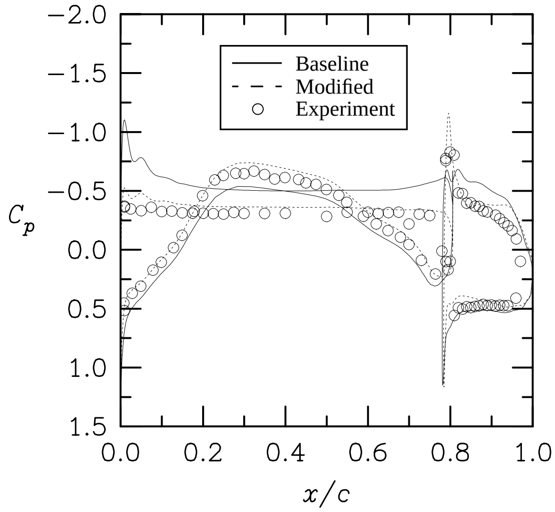

Figure 17. Pressure distributions for multielement airfoil problem.

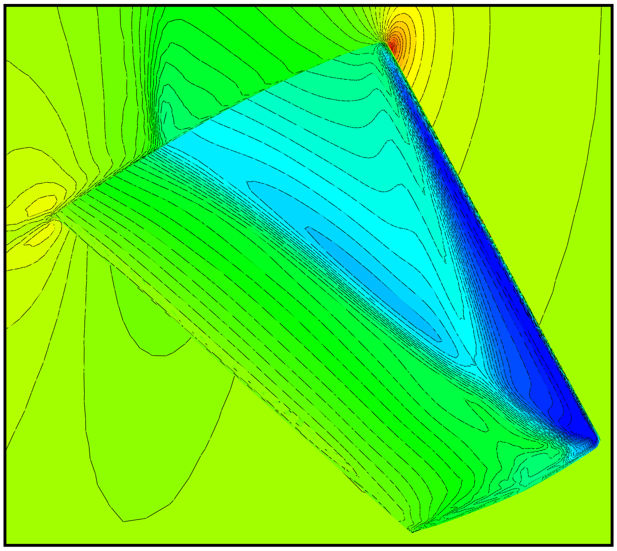

Figure 18. Density contours for the baseline geometry.

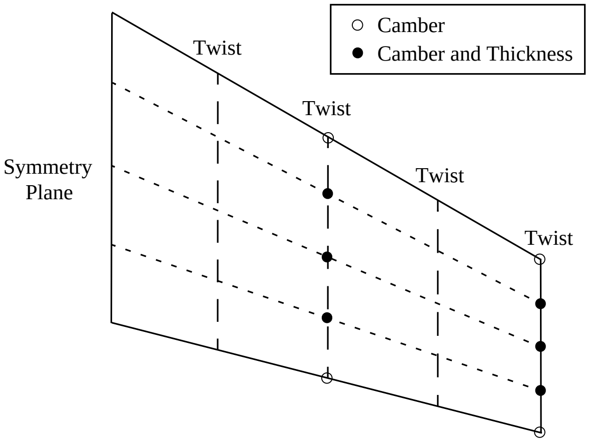

Figure 19. Location of design variables for ONERA M6 wing.

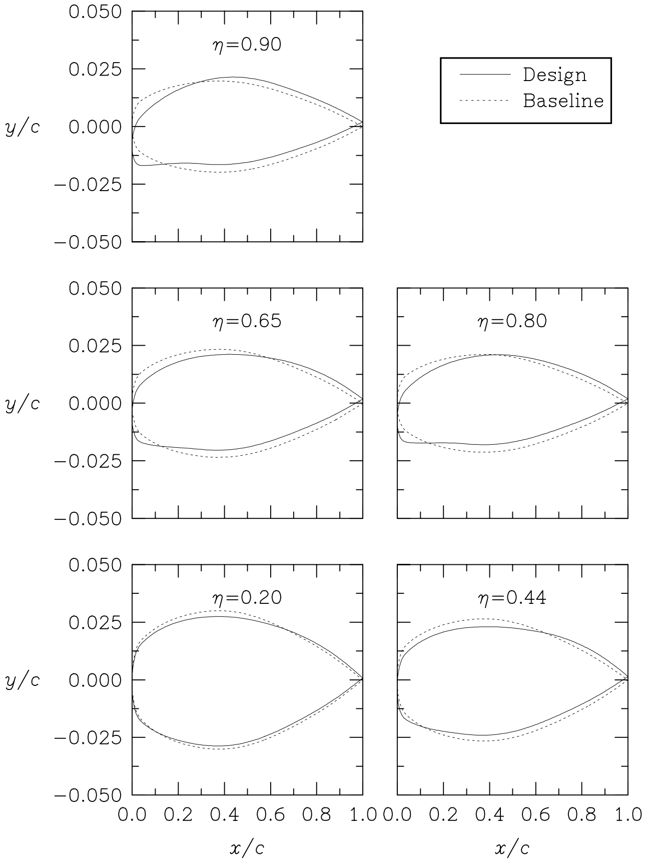

Figure 20. Cross-sections of the initial and final wing geometries.

Turbulent Flow Over ONERA M6 Wing

A turbulent flow wing optimization is performed using the ONERA M6 mesh previously shown in Fig. 4. The freestream Mach number is 0.84, the angle of attack is 3.06°, and the Reynolds number is 5×10^6 based on the MAC. For these conditions, the baseline geometry exhibits a swept shock extending from the root leading edge and a strong normal shock further aft as shown in Fig. 18. The objective for this example is to reduce drag while maintaining a specified lift. The 20 shape design variables are shown in Fig. 19, and the angle of attack is also allowed to vary. The design case has been run using approximately 3 days of wallclock time on 12 processors of an Origin 2000 system.

Cross-sections of the initial and final geometries can be seen in Fig. 20. After 5 design cycles, the drag coefficient has been decreased by 15% from 0.0168 to 0.0142, while the lift coefficient has maintained its baseline value of 0.253. Pressure distributions at several locations across the span of the wing are shown in Fig. 21, and density contours for the final geometry are shown in Fig. 22. It can be seen that the normal shock has been weakened considerably, particularly in the outboard section of the wing.

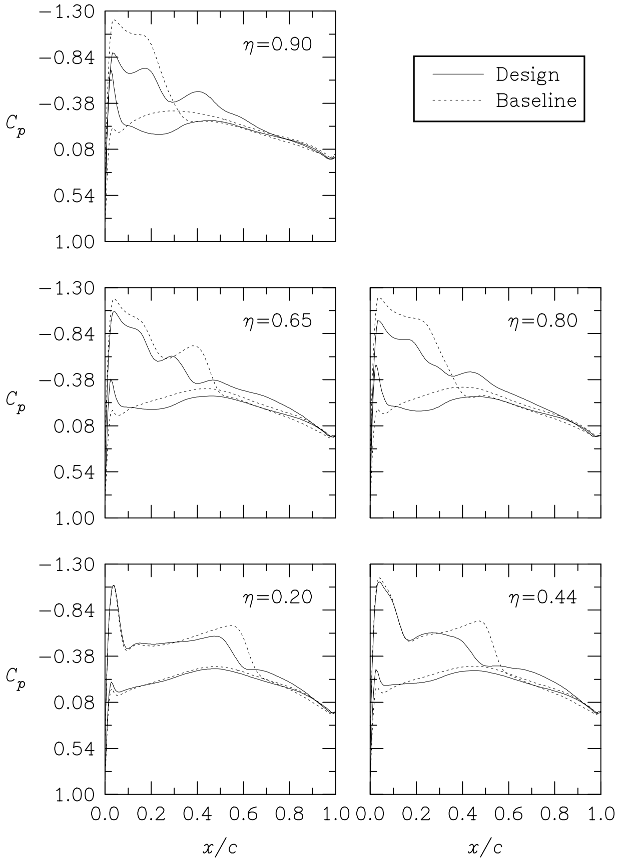

Figure 21. Pressure distributions for the initial and final wing geometries.

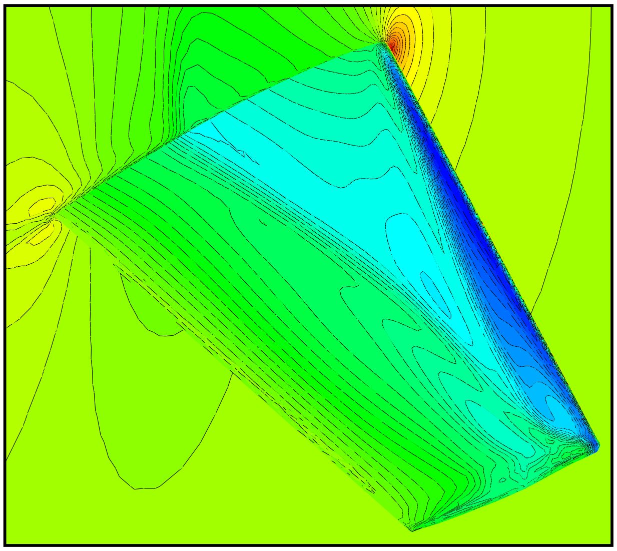

Figure 22. Density contours for the final geometry.

Turbulent Flow Over Multielement Wing

In order to handle an arbitrary number of three-dimensional elements parameterized by the package described in Ref. 24, software

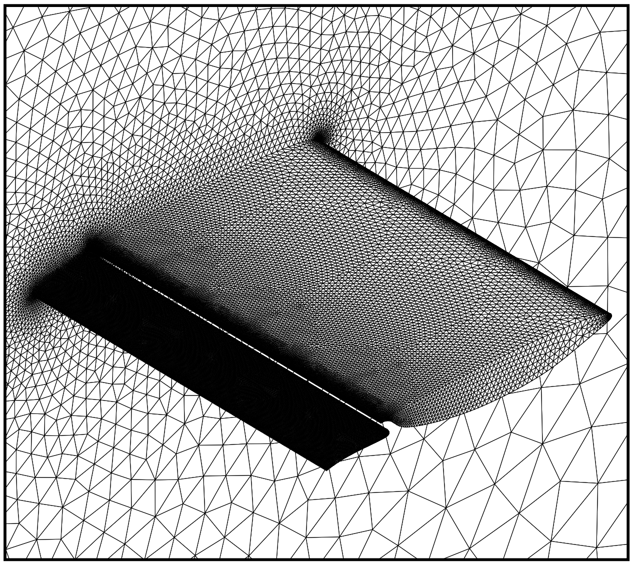

Figure 23. Surface mesh for multielement wing.

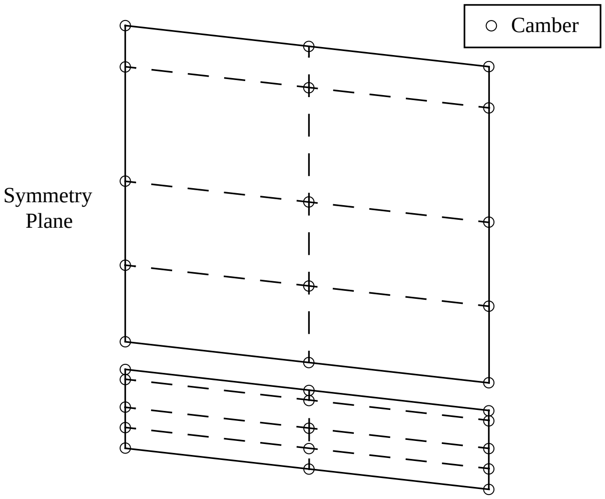

Figure 24. Location of design variables for multielement wing.

has been developed to combine multiple bodies which employ independent parameterizations. To evaluate this capability, the baseline airfoil depicted in Fig. 16 is extruded in a spanwise direction to create a 5° swept wing as shown in Fig. 23. The surface grid shown contains 31,229 nodes and the volume mesh consists of 843,385 nodes and 4,796,360 tetrahedra. The adjoint solver requires roughly 12 gigabytes of storage for the current example.

For this case, the main element and flap are parameterized separately using camber values at the locations shown in Fig. 24. In addition to these shape parameters, the deflection as well as the vertical and streamwise positioning of the flap are used as design variables. The angle of attack is also allowed to vary, for a total of 34 design variables. The objective is to reduce the drag while maintaining a specified lift. The freestream Mach number is 0.75, the baseline angle of attack is 1°, and the Reynolds number is 6.2 million.

The design case has been run using 16 processors of an Origin 2000 system and required approximately 6 days of wallclock time. After 5 design cycles, the drag coefficient has been reduced from 0.0399 to 0.0378, while the lift coefficient has maintained its original value of 0.437. Cross-sections of the baseline and modified geometries can be seen in Fig. 25, while pressure distributions are shown in Fig. 26. The flap has been repositioned and can be seen to carry a reduced loading, which has been compensated by an increase in camber across the main element.

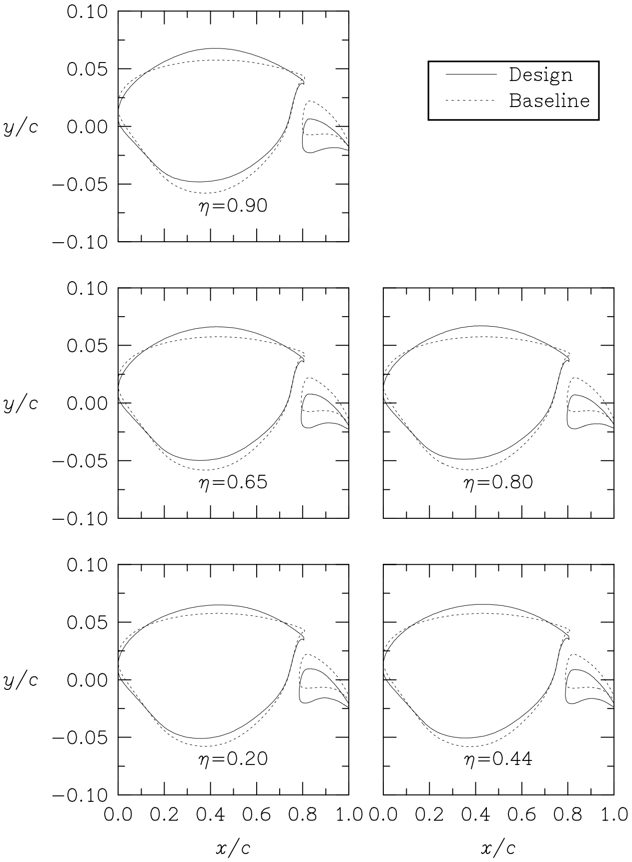

Figure 25. Cross-sections of the initial and final wing geometries.

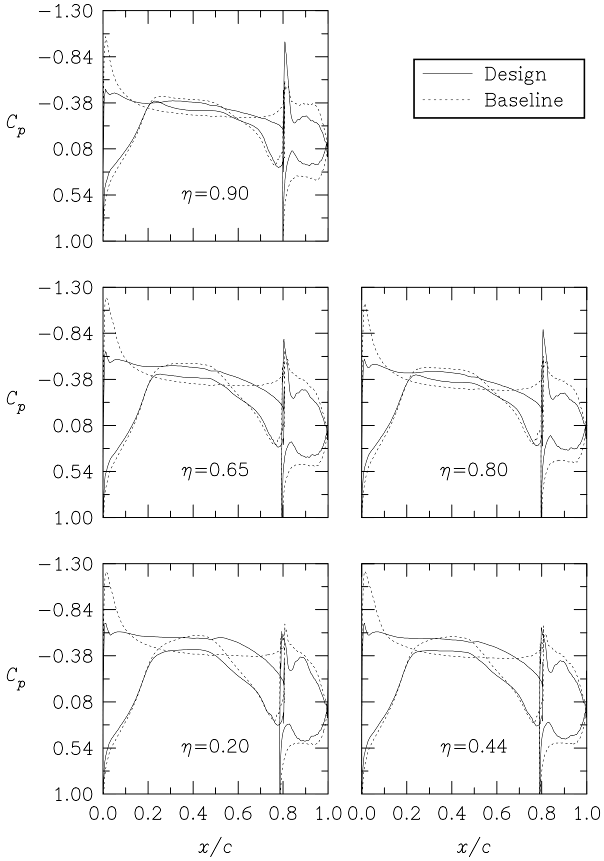

Figure 26. Pressure distributions for the initial and final wing geometries.

Summary

An unstructured mesh design methodology based on a discrete adjoint formulation has been extended to a multiprocessing environment using domain decomposition and a shared memory approach. The parallel implementation has been shown to scale well while yielding discretely consistent sensitivity information.

A preconditioning scheme based on the complete linearization of the residual has been demonstrated for adjoint computations. Although the new strategy requires an increased amount of memory due to the larger stencil, it has been found to give superior convergence rates and hence better reliability.

An improved mesh movement capability has been developed using an approach based on linear elasticity relations. In the current work, the scheme is modified to use a spatially-varying value of Poisson's ratio to account for highly skewed cells. The new procedure yields a robust technique which maintains the mesh topology in a more consistent fashion than a previous distance function/tension-spring analogy.

Several design examples have been presented which demonstrate the improved capability of the current implementation. Reduced turnaround time combined with an increased level of robustness has enabled previously impractical problems to be addressed.

Acknowledgments

The authors would like to thank Jamshid Samareh for continued assistance with his geometric parameterization package and Ben Anders and Sally Viken for the experimental data used in the first design example.

References

[1] Anderson, W.K., and Bonhaus, D.L., “Aerodynamic Design on Unstructured Grids for Turbulent Flows,” NASA TM 112867, June 1997.

[2] Anderson, W.K., and Bonhaus, D.L., "Airfoil Design on Unstructured Grids for Turbulent Flows," *AIAA Journal*, Vol. 37, No. 2, 1999, pp. 185-191.

[3] Nielsen, E.J., and Anderson, W.K., “Aerodynamic Design Optimization on Unstructured Meshes Using the Navier-Stokes Equations,” AIAA Paper 98-4809, September 1998.

[4] Nielsen, E.J., “Aerodynamic Design Sensitivities on an Unstructured Mesh Using the Navier-Stokes Equations and a Discrete Adjoint Formulation,” Ph.D. Thesis, Virginia Polytechnic Institute and State University, 1998.

[5] Anderson, W.K., and Venkatakrishnan, V., “Aerodynamic Design Optimization on Unstructured Grids with a Continuous Adjoint Formulation,” AIAA Paper 97-0643, January 1997.

[6] Jameson, A., Pierce, N.A., and Martinelli, L., “Optimum Aerodynamic Design Using the Navier-Stokes Equations,” AIAA Paper 97-0101, January 1997.

[7] Reuther, J., Alonso, J.J., Martins, J.R.R.A., and Smith, S.C., “A Coupled Aero-Structural Optimization Method for Complete Aircraft Configurations,” AIAA Paper 99-0187, January 1999.

[8] Elliott, J., and Peraire, J., “Aerodynamic Optimization on Unstructured Meshes with Viscous Effects,” AIAA Paper 97-1849, June 1997.

[9] Soemarwoto, B., “Multipoint Aerodynamic Design by Optimization,” Ph.D. Thesis, Delft University of Technology, 1996.

[10] Mohammadi, B., “Optimal Shape Design, Reverse Mode of Automatic Differentiation and Turbulence,” AIAA Paper 97-0099, January 1997.

[11] White, F.M., Viscous Fluid Flow, McGraw-Hill, New York, 1974.

[12] Spalart, P.R., and Allmaras, S.R., “A One-Equation Turbulence Model for Aerodynamic Flows,” AIAA Paper 92-0439, January 1992.

[13] Anderson, W.K., and Bonhaus, D.L., “An Implicit Upwind Algorithm for Computing Turbulent Flows on Unstructured Grids,” Computers and Fluids, Vol. 23, No.1, 1994, pp. 1-21.

[14] Anderson, W.K., Bonhaus, D.L., McGhee, R., and Walker, B., “Navier-Stokes Computations and Experimental Comparisons for Multielement Airfoil Configurations,” AIAA Journal of Aircraft, Vol. 32, No. 6, 1995, pp. 1246-1253.

[15] Roe, P.L., "Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes," *J. Comp. Phys.*, Vol. 43, No. 2, 1981, pp. 357-372.

[16] Van Leer, B., "Flux Vector Splitting for the Euler Equations," Lecture Notes in Physics, Vol. 170, 1982, pp. 501-512.

[17] Chakravarthy, S.R., and Osher, S., "Numerical Experiments with the Osher Upwind Scheme for the Euler Equations," *AIAA Journal*, Vol. 21, No. 9, 1983, pp. 1241-1248.

[18] Marcum, D.L. "Generation of Unstructured Grids for Viscous Flow Applications," AIAA Paper 95-0212, Jan. 1995.

[19] Pirzadeh, S., "Viscous Unstructured Three-Dimensional Grids by the Advancing-Layers Method," AIAA Paper 94-0417, Jan. 1994.

[20] Saad, Y., and Schultz, M.H., "GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems," *SIAM J. Sci. Stat. Comput.*, Vol. 7, 1986, pp. 856-869.

[21] Hackbusch, W., *Iterative Solution of Large Sparse Systems of Equations*, Springer-Verlag, New York, 1994.

[22] Karypis, G., and Kumar, V., “A Fast and High Quality Multilevel Scheme for Partitioning Irregular Graphs,” *SIAM Journal of Scientific Computing*, Vol. 20, No. 1, 1998, pp. 359-392.

[23] `http://www.openmp.org`

[24] Samareh, J., “Geometry Modeling and Grid Generation for Design and Optimization,” ICASE/LaRC/NSF/ARO Workshop on Computational Aerosciences in the 21st Century, April 22-24, 1998.

[25] Barth, T. "Parallel CFD Algorithms on Unstructured Meshes," Paper presented in an AGARD-FDP-VKI special course on "Parallel Computing in CFD," held at the Von Karman Institute, Rhode-Saint-Genese, Belgium, from 15-19 May 1995 and 16-20 October 1995 at NASA Ames, United States and published in R-807.

[26] Pilkey, W.D., and Wunderlich, W., Mechanics of Structures: Variational and Computational Methods, CRC Press, Florida, 1994.

[27] Johnson, A.A., and Tezduyar, T.E., "Simulation of Multiple Spheres Falling in a Liquid-Filled Tube," *Comput. Methods Appl. Mech. Engrg.*, Vol. 134, 1996, pp. 351-373.