AIAA SciTech
5-9 January 2015, Kissimmee, Florida
53rd AIAA Aerospace Sciences Meeting

Adaptive Optimization-Based Smoothing for Tetrahedral Meshes

Steve L. Karman*
University of Tennessee
Chattanooga, Tennessee, 37403

A tetrahedral mesh smoothing method is presented that uses an optimization-based node perturbation technique to minimize a cost function. The cost function is based on the element condition number of the surrounding tetrahedra. This condition number incorporates a weight matrix that transforms the element from physical space to a reference space. Feature-based adaptation is incorporated into the element weight matrix to influence the node perturbation scheme to favor non-unit aspect ratio elements. The smoothing scheme produces meshes that cluster grid points in high gradient regions of the selected adaptation function. Results for several inviscid, three-dimensional cases are included.

Nomenclature

A	=	Jacobian matrix for condition number
A_c	=	Adaptation coefficient
AF	=	Adaptation function
C, C_e	=	Element cost
CN	=	Element condition number
d_0-d_5	=	Relative length of edge
h_{min}	=	Minimum relative edge length
J	=	Jacobian
$\vec{p}_n$	=	Perturbation vector for node n
$\vec{s}_n$	=	Sensitivity vector for node n
W	=	Weight matrix for condition number
X, Y, Z	=	Cartesian physical coordinates
∇f	=	Gradient of selected function

I. Introduction

Three-dimensional unstructured meshes composed of tetrahedra are extremely popular in computational fluid dynamics due to the ease of construction. Complicated geometry can be meshed with relative ease with the current state of the art tetrahedral mesh generation methods. Several different approaches are employed in the creation of the meshes, such as Delaunay and Advancing Front methods.[1-7] These generation methods typically produce valid meshes that predominantly contain high quality elements.

Mesh quality can be measured in a number of different ways. Measures such as aspect ratio, minimum and maximum included angles and element condition numbers are some of the metrics used to evaluate the mesh quality. Each mesh generation method has different strengths and weaknesses, resulting in slightly different mesh quality as measured by the various metrics. Smoothing is frequently required to improve the quality of the mesh for the flow simulation. Poor quality can arise from the initial mesh generation process or come about through dynamic movement of boundaries in the mesh. Tetrahedral mesh smoothing is performed using various

* Professor and AIAA Associate Fellow.

techniques including simple Laplacian-type node averaging, tension/torsional spring analogy methods, optimization-based node perturbation methods and some evolving elliptic smoothing methods for unstructured meshes.[8-16] In a typical application all of the mesh generation and smoothing operations take place before any flow solution has been attempted. The quality of the mesh is measured using industry accepted quality metrics that are based on geometric information in the mesh. No flow solution information is used.

Mesh adaptation uses information from a flow solution to determine where and how to modify the existing mesh. The modification can be in the form of refinement and coarsening of the mesh or mesh movement. Refinement and coarsening techniques are categorized as feature based or output (Adjoint) based.[17-21] Mesh movement methods seek to reposition the nodes to better resolve high gradient regions of an adaptation function such as velocity magnitude, pressure or Mach number.[22, 23] In either case the goal is to alter the spacing in the mesh to improve the resolution of pertinent features in the simulation.

This paper describes an extension of an optimization-based node perturbation method that allows for feature-based adaptation. The method seeks to reposition each node to minimize local cost functions. The cost functions consider element shape and gradients of an adaptation function. Throughout this paper the original mesh connectivity is maintained. Neither refinement nor deletion is used. Also, no mesh reconnection through edge/face flips occurs. The mesh is adapted solely through node repositioning with a constant connectivity. Details of the baseline node perturbation scheme are included. The extension of the method for adaptation is then provided. Results are shown for several complex 3D configurations and flowfields. Initial tetrahedral meshes were generated using Pointwise.[24] All visualizations were generated using FieldView.[25]

II. Optimization-Based Mesh Smoothing

Optimization techniques seek to optimize some cost function related to the desired output. Mesh smoothing techniques strive to improve mesh quality as defined by various geometric quality measures. Therefore, optimization-based mesh smoothing seeks to optimize mesh quality as defined by a cost function based on a given geometric quality metric. The basic optimization-based mesh smoothing will first be described. This will be followed by the extension of the method for feature-based adaptation.

The basic optimization-based mesh smoothing method was inspired by the published work of Freitag and Knupp.[10, 12] The main differences between the method described by Freitag and Knupp are in the definition of a unified cost function and the method for perturbing the nodes of the mesh to minimize the cost. A cost function is defined for each element based on the status of the element. If the element Jacobian, J, is negative or zero the cost function will be greater than or equal to one. If the element is not inverted the cost is based on the element condition number, CN, and will be less than one. Equation (1) shows the two version of the element cost function, C. The reason for two formulations is because the condition number does not recognize when the element is inverted. The unification of the two states allows the optimizer to handle inverted elements in a more natural way. This formulation is C0 continuous at the value of one.

$$C = 1 - J, \text{ if J <= 0.0}$$
$$C = 1 - \frac{1}{CN}, \text{ if J > 0.0} \quad (1)$$

The second form of the cost function is based on the weighted element condition number, CN. For tetrahedra, the weighted condition number is given below in Equation (2).

$$CN = \frac{\left\|AW^{-1}\right\|\left\|WA^{-1}\right\|}{3} \tag{2}$$

The bracketed quantities are the Frobenius norms of the matrix product. The A matrix is Jacobian matrix consists of columns using the three edge vectors emanating from a corner of the tetrahedron, shown in Figure 1 and defined in Equation (3).

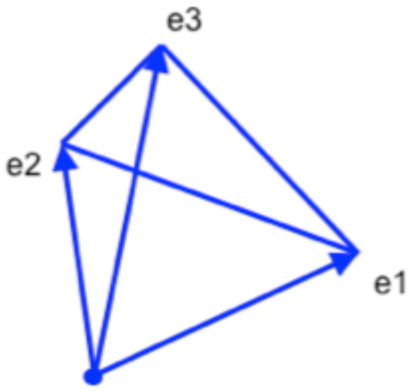

Figure 1. Edge vectors for Jacobian matrix.

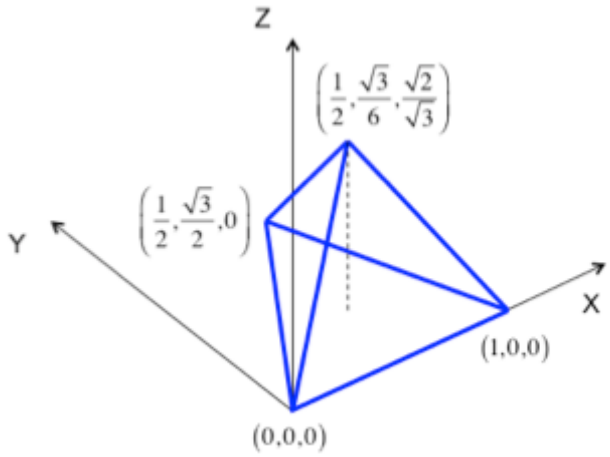

Figure 2. Vertex location for ideal tetrahedron.

The W matrix is the weight matrix that transforms the reference tetrahedron (a right-angled tetrahedron) to a regular tetrahedron, shown in Figure 2. A regular tetrahedron is one in which all four faces are equilateral triangles, sometimes referred to as an equilateral tetrahedron.

$$A = \begin{bmatrix} e1_x & e2_x & e3_x \\ e1_y & e2_y & e3_y \\ e1_z & e2_z & e3_z \end{bmatrix}$$

$$W = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & \frac{\sqrt{3}}{6} \\ 0 & 0 & \frac{\sqrt{2}}{\sqrt{3}} \end{bmatrix} \tag{3}$$

The value of the weighted condition number will be unity for an equilateral tetrahedron and will increase towards infinity for a flattened tetrahedron. The cost function, C, will vary from zero for an equilateral tetrahedron and approach one for a collapsed tetrahedron. At that point the Jacobian based formula is used and the cost will increase with increasing values of the negative Jacobian.

Frietag and Knupp used multiple search directions and function evaluations to determine the new position of each node.[10] The stated reason was the inability to use gradient-based optimization due to discontinuities in the objective function and singularities where elements collapse. Their objective was to find a new node position based on a norm of the condition

number of the surrounding elements; i.e. the L2 or L∞ norm of the element condition numbers. Previous work by this author experienced similar behavior, even though the cost function in Equation (1) combined the condition number and Jacobian resulting in a continuous function. The node-based cost function was too erratic to compute meaningful gradient information.

In the new approach nodes are perturbed in directions consistent with the sensitivity of the element cost function with respect to each node of the element. These element-based sensitivities are well behaved for the derived cost function for an element. The red triangle in Figure 3 has a high cost function as it is near collapse. The vectors emanating from the corners of the triangle are the sensitivity vectors for each node. They are the gradient vectors for each node for increasing the cost of the element. The black triangle represents the ideal element. Perturbation directions for each node are simply the reverse of the gradient vectors (i.e. move the nodes in the opposite direction of the sensitivity).

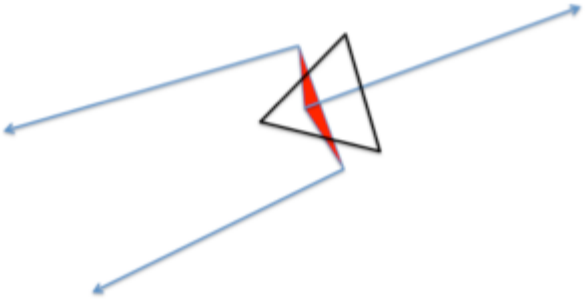

Figure 3. Sensitivity vectors for element cost function.

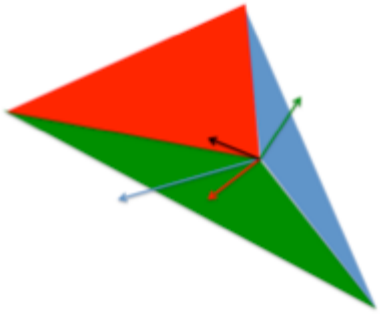

Figure 4. Combined perturbation vectors for central node.

A nodal perturbation vector is constructed by a cost-weighted average of the sensitivity vector for that node from all the elements surrounding the node, defined in Equation (4). The sensitivity vector for the node for each element is given by $\vec{s}_n$. These are weighted by the element cost function for the number of elements surrounding the node, *ne*. In Figure 4 the element-based perturbation vectors for the central node are color-coded by the element color. The black vector represents the weighted average of the element-based perturbation vectors. Since the weight used is the element cost, the higher cost elements will have more influence. As the cost decreases the magnitude of the gradient vectors will also decrease. The cost-weighted average vector will typically be smaller in magnitude than the maximum element-based vectors. The perturbation amount is restricted to a fraction of the smallest distance to any neighboring node, typically 1% to 5%. The magnitude of the average perturbation vector is also limited to one or less. Multiple iterations are required to "converge" the mesh to a stable configuration.

$$\vec{p}_n = -\frac{\sum_{e=1}^{ne} \left(\vec{s}_n\right)_e C_e}{\sum_{e=1}^{ne} C_e} \tag{4}$$

The sensitivities can be computed a number of ways. Differentiating the cost function formulae with respect to the X, Y and Z coordinates of the four corner nodes generates the analytic derivatives. This would require full expansion of the matrix products and squares of the elements in the Frobenius norms. If finite differences are used then 13 evaluations of the cost are required; one for the base cost value and one for changes in each coordinate at the four corners of the tetrahedron. An epsilon value for the coordinate change would be required that was small enough to produce valid numerical derivatives. If the epsilon value is too small the finite-differences could suffer from subtractive cancellation errors. Complex Taylor Series Expansions could be used, thereby eliminating subtractive cancellation errors, but would still require 13 function evaluations.[26]

The chosen method for computing the sensitivities uses C++ function overloading to compute the function value and the twelve derivative quantities through numerical chain rule differentiation.[27, 28] A single cost function call is required for each element. The function call returns the cost for the element and the sensitivities for each of the four corner coordinates. Experience has shown the computational cost of the function overloading approach to be comparable to the complex Taylor series approach.

III. Adaptive Weight Matrices

The ideal weight matrix given in Equation (3) will drive the mesh toward regular tetrahedra (tetrahedra composed of equilateral triangular faces). In two dimensions it is extremely rare to have a mesh with all equilateral triangles. The perfect two-dimensional triangular mesh would be a tiling of space using equilateral triangles. This situation is extremely rare. Perfect equilateral tetrahedra cannot be used to fill three-dimensional space. Space can be filled with regular tetrahedra and regular octahedra, but not with regular tetrahedra alone. So the cost in three dimensions can never be zero for every element when optimizing a mesh for quality.

The reference element can be redefined to allow non-equilateral shapes. If the lengths of the six edges are defined then the coordinates of the reference element can be determined. Given the 6 edges lengths (d0-d5) the ideal element node coordinates are shown in Figure 5. This assumes that a valid set of edge lengths are defined. The edge lengths for any given face of the tetrahedron must form a valid triangle. Consider the bottom face composed of edges 0, 1 and 2. If the lengths were defined as shown in the top of Figure 6 an invalid triangle results. In the middle of the figure a valid triangle normalized by maximum edge length is shown with a user defined minimum height, h_{min}. The perimeter of this triangle is $2 + h_{min}$. In the bottom of the figure edges d1 and d2 are modified to conform to a scaled version of the total perimeter distance. Any set of defined edge lengths for a face of the tetrahedron must conform to this minimum perimeter distance rule, i.e. valid triangular faces are produced in the reference element.

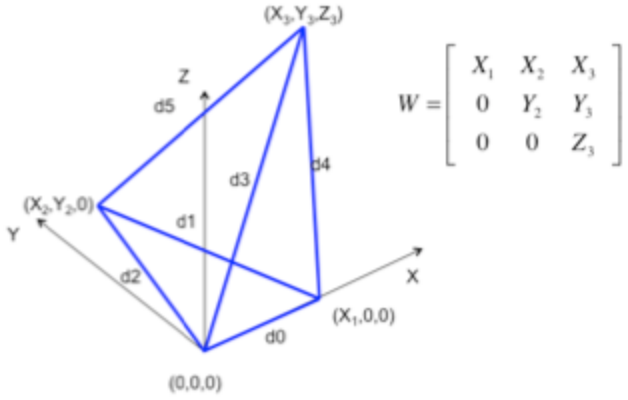

Figure 5. Coordinates for weight matrix with defined edge lengths.

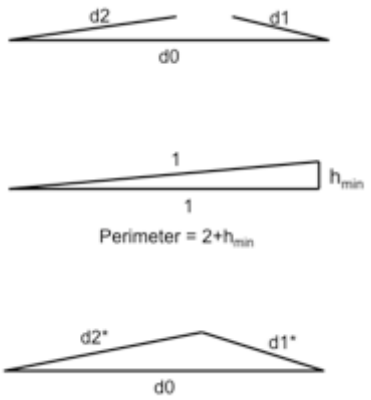

Figure 6. Edge lengths modified to produce valid triangle.

Given a valid set of edge lengths, d0 – d5, then the unknown coordinate values for the reference element are computed using Equations (5). A test of the formulae is to supply the actual edge lengths from a given element in the mesh. In this case the resulting weighted condition number is one and the cost is identically zero.

$$
\begin{aligned}
X_1 &= d_0 \\
X_2 &= \frac{d_0^2 + d_2^2 - d_1^2}{2d_0} \\
Y_2 &= \sqrt{d_2^2 - X_2^2} \\
X_3 &= \frac{d_0^2 + d_3^2 - d_4^2}{2d_0} \\
Y_3 &= \frac{X_2^2 + Y_2^2 - X_2\left(d_0 + \frac{d_3^2}{d_0} - \frac{d_4^2}{d_0}\right) + d_3^2 - d_5^2}{2Y_2} \\
Z_3 &= \sqrt{d_3^2 - X_3^2 - Y_3^2}
\end{aligned}
\tag{5}
$$

The edge lengths provided to the weight matrix are relative edge lengths. They do not need to match the physical mesh scales. They define an element shape, just as the original weight matrix did for a regular tetrahedron. If all edge lengths are defined to be unity the original weight matrix is produced.

For any given mesh there exists a valid set of defined edge lengths that result in all element costs equal to zero. The task is to define a new set of relative edge lengths for a mesh that influence the optimizer to cluster the mesh to features in the flowfield solution. This process is feature-based adaptation, so the desired shapes are dependent on gradients of a function such as pressure, velocity magnitude or Mach number. Relative lengths for each edge in the mesh are computed based on gradients of the chosen adaptation function, depicted in Figure 7 and Figure 8. The gradient of the adaptation function for an edge is the average of the nodal gradient vector.

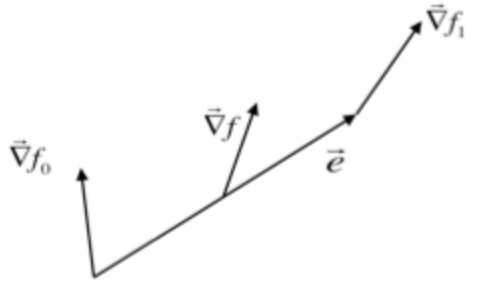

Figure 7. Function gradient vector for edge is average of nodal gradient vectors.

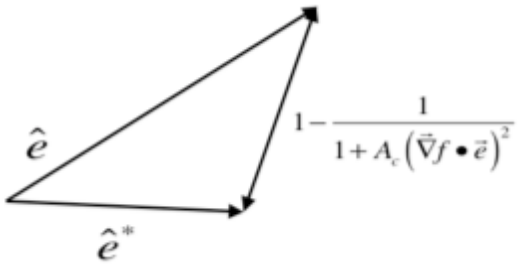

Figure 8. Relative edge vector is reduced in direction of gradient.

The normalized edge vector, $\hat{e}$, is reduced in the direction of the gradient by a factor based on the square of the dot product of the edge vector and function gradient vector. If the gradient is zero or the gradient is perpendicular to the edge then no reduction in length occurs and the relative length is one. The user can control the amount of adaptation using the coefficient A_c. Each edge in the mesh is processed in this manner. The basic idea is to squeeze the elements in the direction of the gradient of the adaptation function. Enforcement of the relative perimeter distance is checked for each triangle face of each element.

The use of the dot product of the gradient vector and the edge vector results in reduction factors directly related to function value changes in the direction of the edge. When discontinuities, such as shocks, occur this factor will approach a constant value as the mesh is squeezed. Where there are no gradients in the adaptation function the original optimization scheme is recovered. The adaptation coefficient, A_c, can be directly specified or automatically computed using the maximum gradient magnitude and the specified h_{min} parameter.

IV. Results

The flow solver used for all cases is a finite-volume, node-centered Euler code that uses Roe's scheme for inviscid fluxes. The code uses an implicit BDF1 or BDF2 time integration scheme and second order accurate spacial discretizations.[29] It supports tetrahedral elements only. Possible adaptation functions include density, pressure, velocity magnitude and Mach number.

A. Analytic Shock Function

Anderson performed mesh adaptation on several two-dimensional analytic cases.[22] One case was a shock-like function defined using Equations (6). Elliptic smoothing with adaptation was performed on a structured two-dimensional mesh with dimensions 24 X 24.

$$u = \begin{cases} 1.0 & 0 \le x \le 0.5(y+10) \\ 0.5(7-x)+0.25y, & 0.5(y+10) \le x \le 0.5(y+14) \\ 0.0 & 0.5(y+14) \le x \le 24 \end{cases} \tag{6}$$

A three-dimensional tetrahedral mesh was created by diagonalizing a structured mesh, shown in Figure 9. The original Z max face for the three-dimensional mesh is shown in Figure 10.

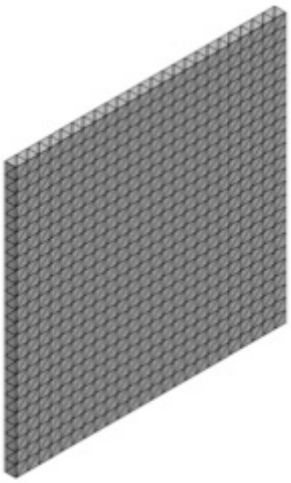

Figure 9. Three-dimensional tetrahedral mesh for shock-like function case.

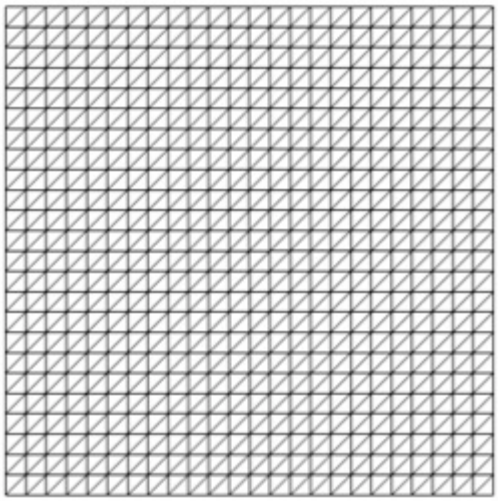

Figure 10. Original diagonalized mesh on constant Z max plane.

Pure optimization smoothing of the mesh without adaptation was performed, resulting in the mesh shown in Figure 11. The optimization method is attempting to modify the mesh toward all equilateral tetrahedra. This is obviously not possible and the resulting mesh tends to cluster toward the upper-left and lower-right corners and away from the upper-right and lower-left corners. This is due solely to the connectivity of the mesh.

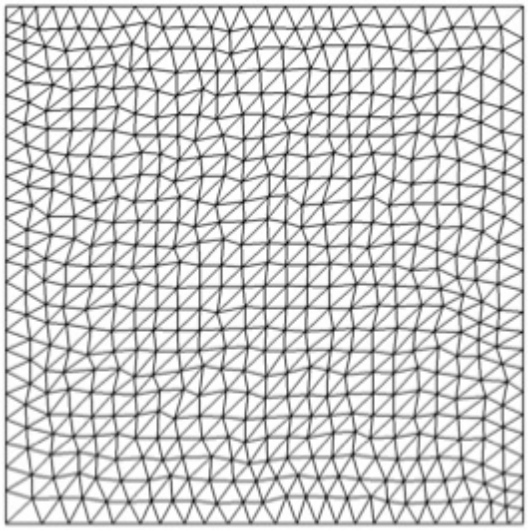

Figure 11. Pure optimization-based smoothing without adaptation.

The adapted tetrahedral mesh is shown in Figure 12. The result from Anderson is shown in Figure 13. The clustering is similar between the two methods. The elliptic method using a structured mesh is slightly smoother. Whereas the optimized unstructured mesh seeks to produce "idealized" tetrahedra according to the modified weight matrix. A tug-of-war of sorts is taking

place between elements in the optimizer approach. Complete convergence is never completely achieved, especially with fixed element connectivity.

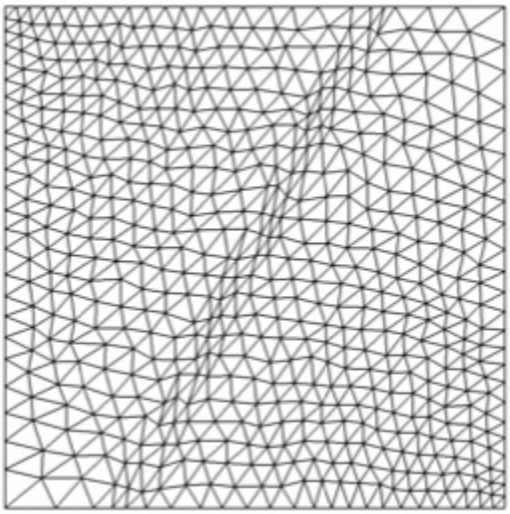

Figure 12. Adapted mesh using the optimization method.

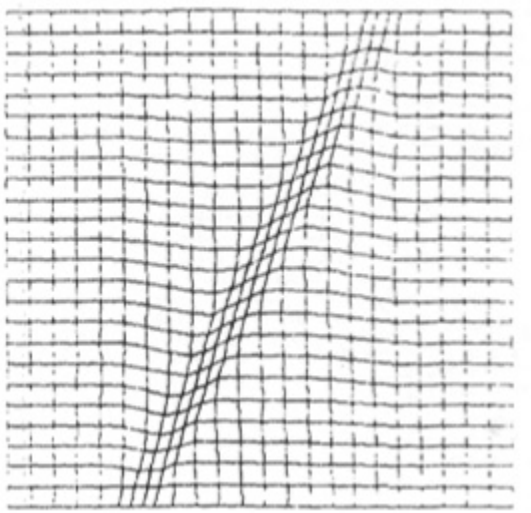

Figure 13. Anderson's adapted mesh for the shock-like function.

A higher resolution tetrahedral mesh was created and adapted to the shock-like function. The optimized mesh without adaptation is shown in Figure 14. The adapted mesh is shown in Figure 15. This result appears much smoother than the coarser mesh result.

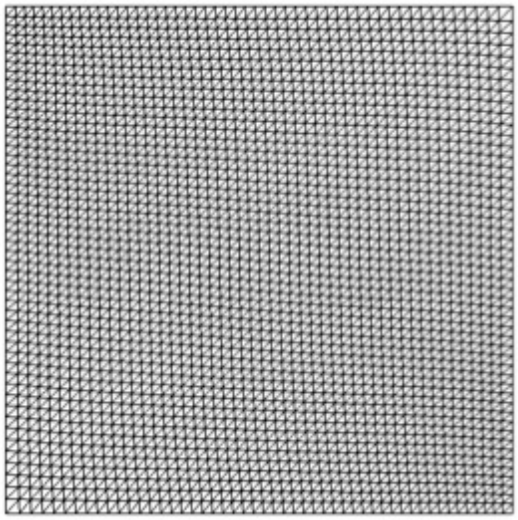

Figure 14. Optimized high-resolution mesh without adaptation.

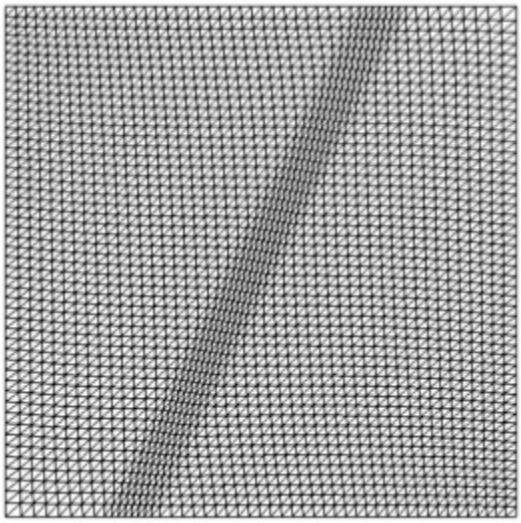

Figure 15. Adapted high-resolution mesh to shock-like function.

B. Analytic Sine Function

Another case shown in Anderson's article included a sine wave function, Equation (7). The same three-dimensional mesh was adapted to this function, shown in Figure 16 compared to Anderson' solution shown in Figure 17. Similar clustering is again achieved in the optimized mesh compared to the structured mesh elliptic result.

$$u = \begin{cases} 0.0, & 0 \le y \le 11 + 4\sin(\pi x/12) \\ 0.5[y - 11 - 4\sin(\pi x/12)], & 11 + 4\sin(\pi x/12) \le y \le 13 + 4\sin(\pi x/12) \\ 1.0, & 13 + 4\sin(\pi x/12) \le y \le 24 \end{cases} \tag{7}$$

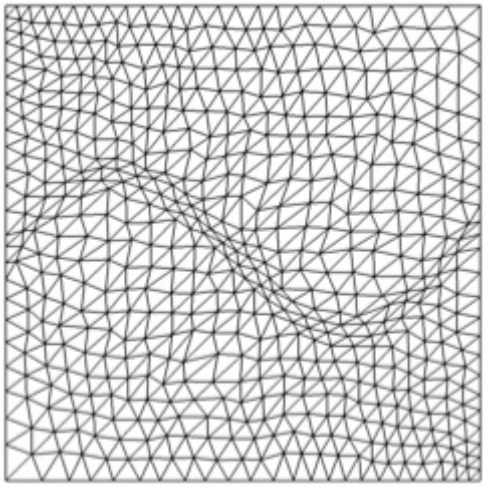

Figure 16. Adapted mesh for sine wave function.

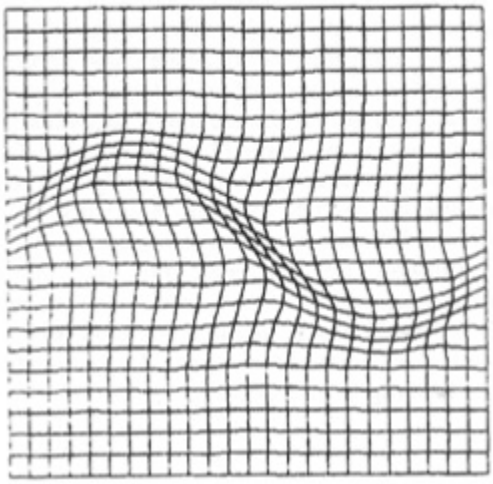

Figure 17. Anderson's adapted mesh for sine wave function.

The higher resolution mesh was adapted with the sine wave function, shown in Figure 18.

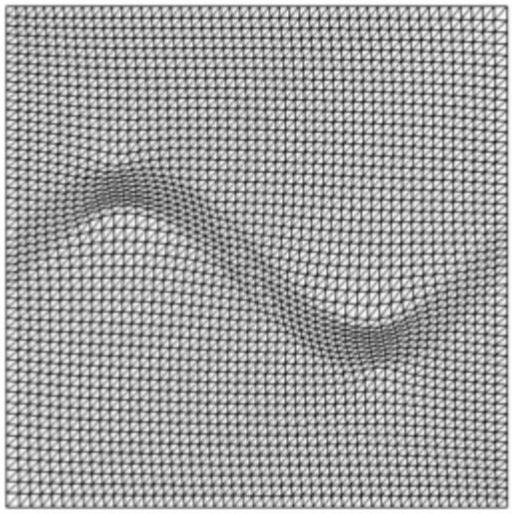

Figure 18. Adatped high resolution mesh for sine wave function.

C. Supersonic Channel Flow

Mach 2 inviscid flow through a converging-diverging channel was the first case adapted using actual flowfield data. This inviscid case is feature rich with multiple shock reflections off the top and bottom boundaries and expansion fans from two corners on the bottom boundary. The initial mesh was generated using Pointwise and is shown in Figure 19 and Figure 20. The geometry and flow conditions can be modeled in two-dimensions but is solved in three dimensions using a tetrahedral mesh.

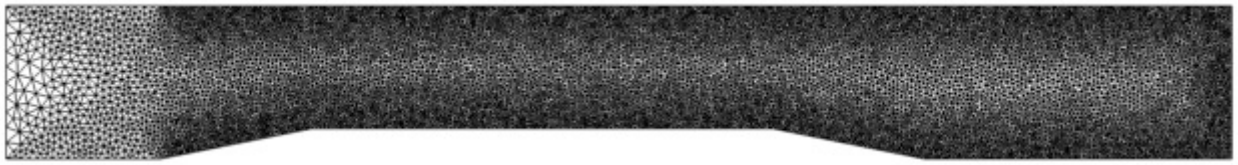

Figure 19. Original surface triangulation on maximum Z boundary.

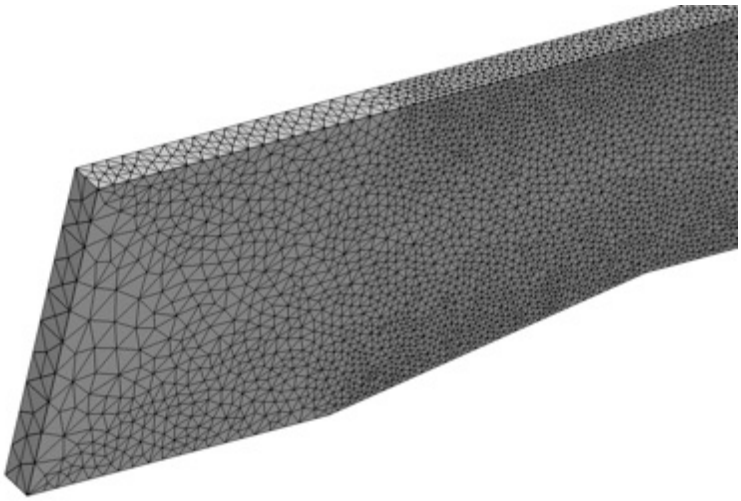

Figure 20. Perspective view showing three-dimensionality of the domain.

This initial mesh was optimized without adaptation and analyzed with the Euler solver. The results are shown in Figure 21. The smoothed mesh shown in the bottom portion of the figure is different from the original, but not noticeable to the eye compared to Figure 19. The Mach number contours shown in the upper portion of the figure show the reflecting shock pattern and expansion fans.

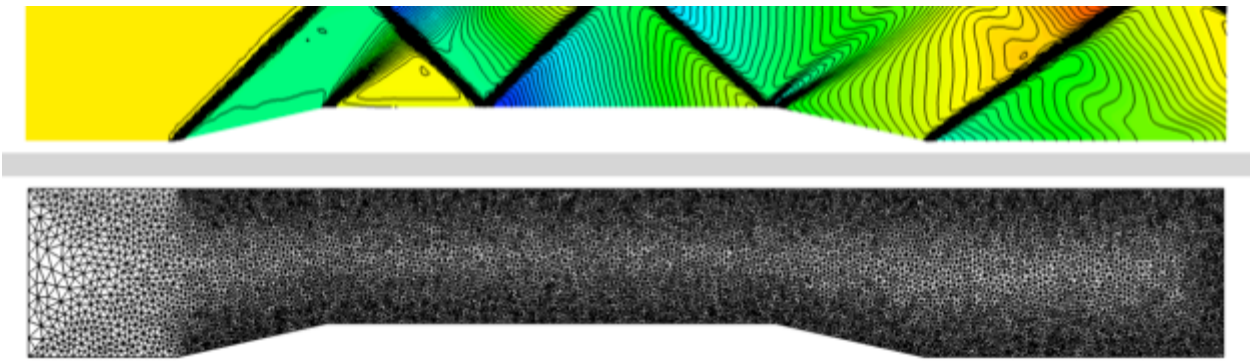

Figure 21. Optimized without adaptation (bottom) and Mach number contours (top).

The final adapted mesh and Mach number contours are shown in Figure 22. The shock structure and expansion fans are evident in the plot of the mesh. The Mach number contours show much crisper shock capturing resulting from the adaptation, to the point where the final reflection from the top boundary is much better resolved.

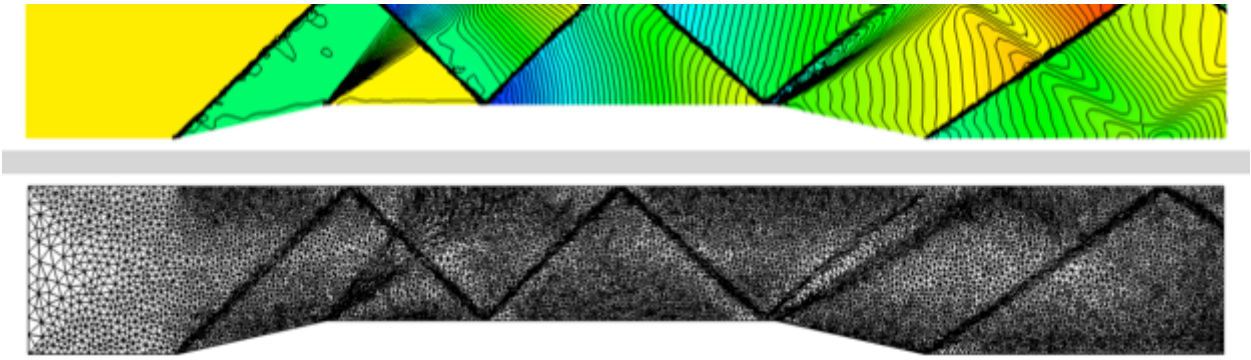

Figure 22. Adapted mesh (bottom) for supersonic channel flow Mach number contours (top).

D. Inviscid Onera M6 Wing

The Onera M6 wing is a well-known CFD validation case that exhibits a complex lambda shock structure on the upper surface.[30, 31] The conditions are inviscid Mach 0.8395 flow at an angle of attack of 3.06 degrees. A comparison of the surface meshes between adapted and optimized is shown in Figure 23. Figure 24 shows just the surface mesh where the clustering of points to the shocks is evident. The adaptation coefficient was automatically computed.

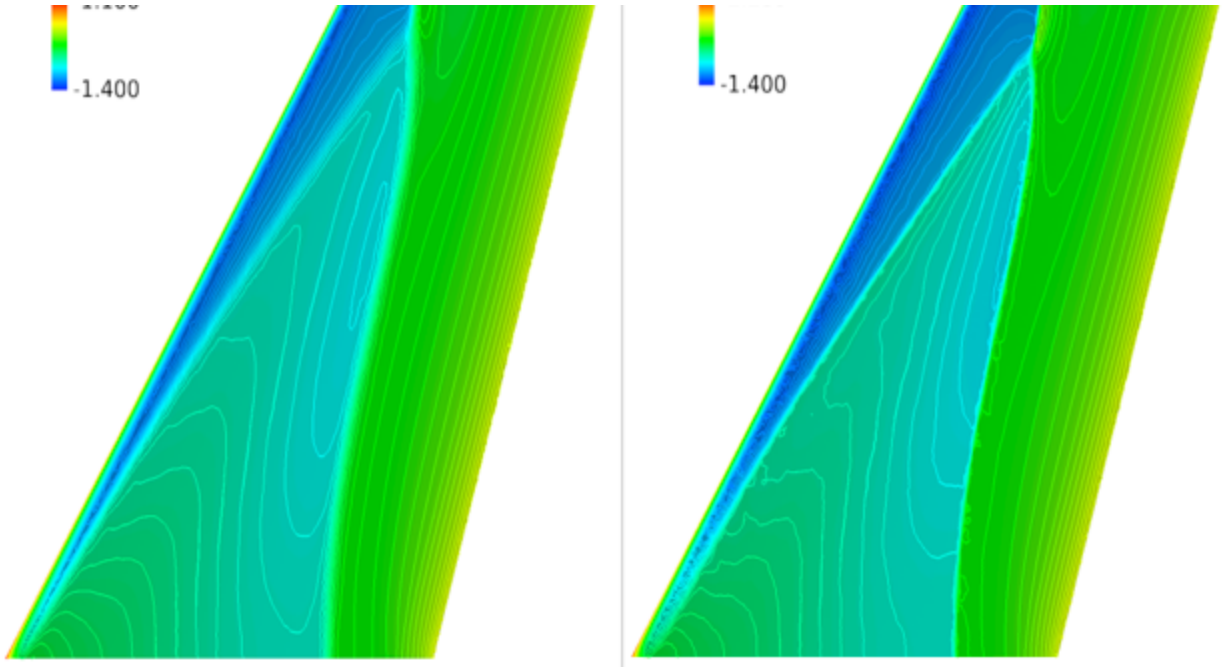

Figure 23. Optimized mesh Cp contours on left and adapted mesh Cp contours on right.

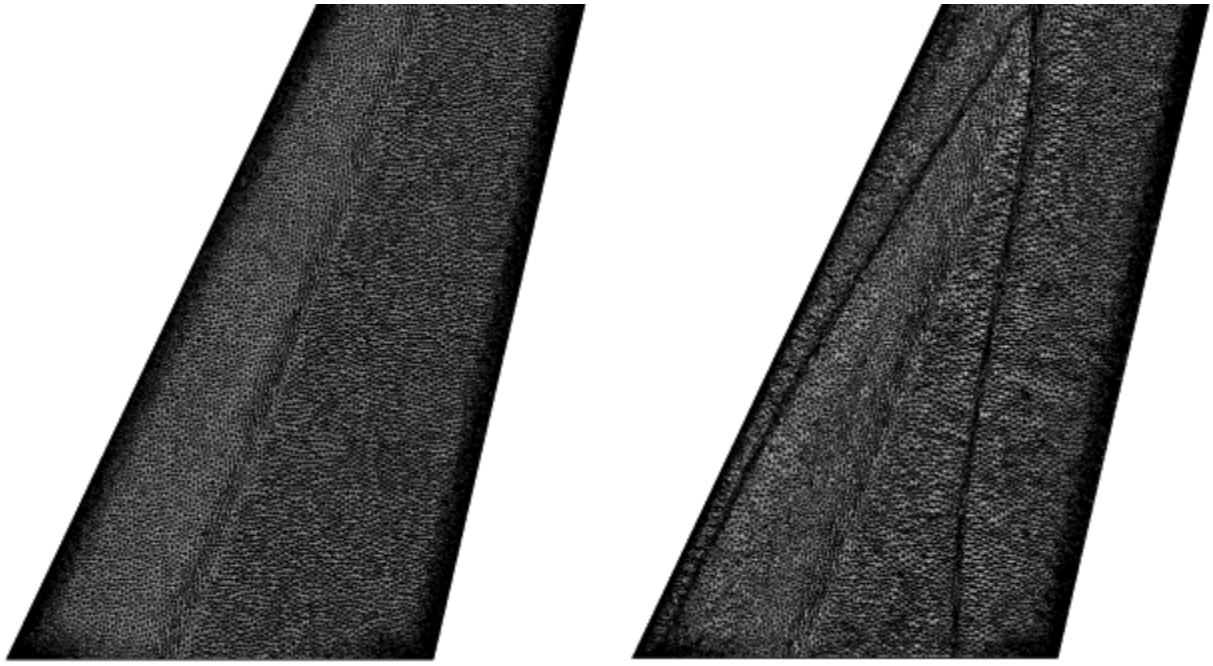

Figure 24. Adapted mesh on right shows clustering toward shock while optimized mesh on left is smooth everywhere.

A comparison of crinkle cuts through the mid-span station is shown in Figure 25. The adaptation is evident at both shock locations on the upper surface. A magnified view of the wing surface mesh is shown in Figure 26.

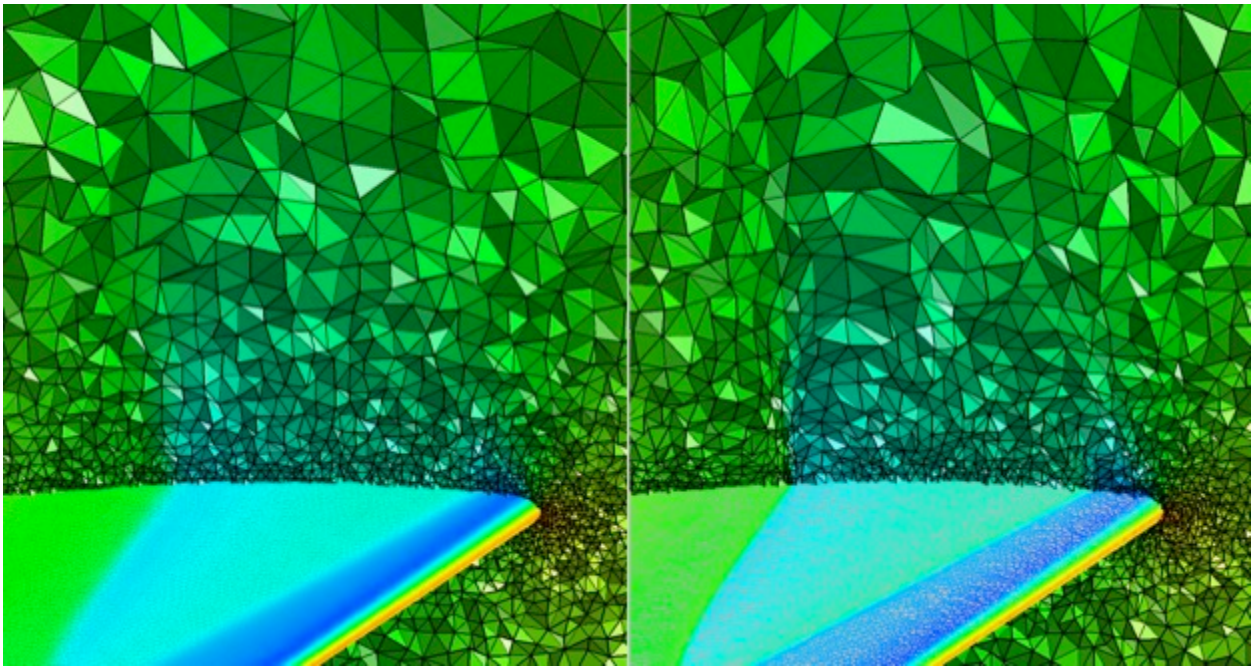

Figure 25. Optimized mesh solution on the left and adapted mesh solution on the right.

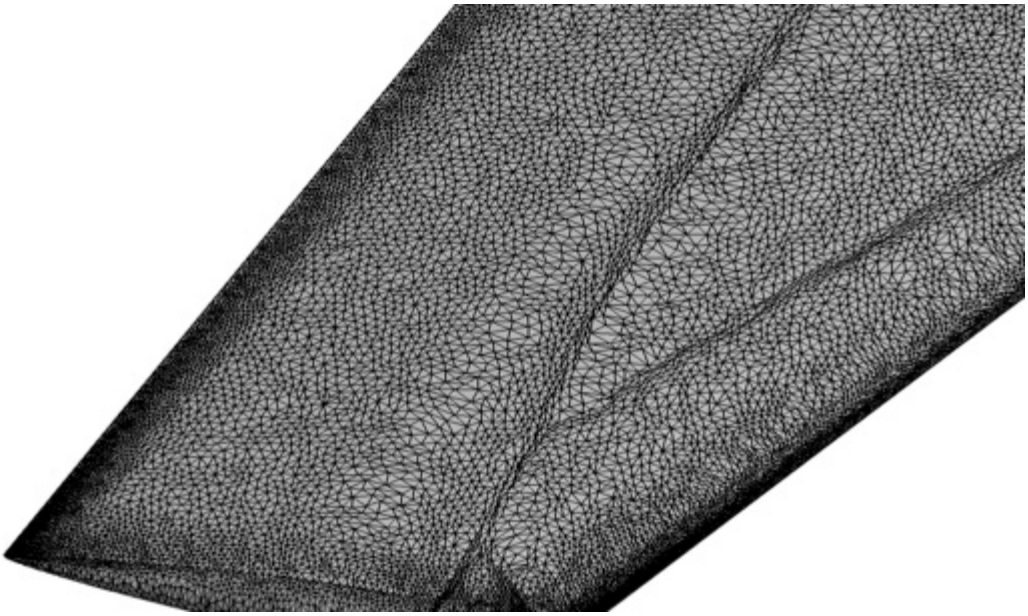

Figure 26. Magnified view of surface mesh near the wing tip.

Surface pressure coefficients are compared between the two solutions and with experimental data in Figure 27 through Figure 32 for span stations 1-6, respectively. In all cases the shocks are more crisply resolved in the adapted mesh solution. The shock locations are consistent with other published inviscid solutions. Viscous effects account for the differences in the axial shock location.

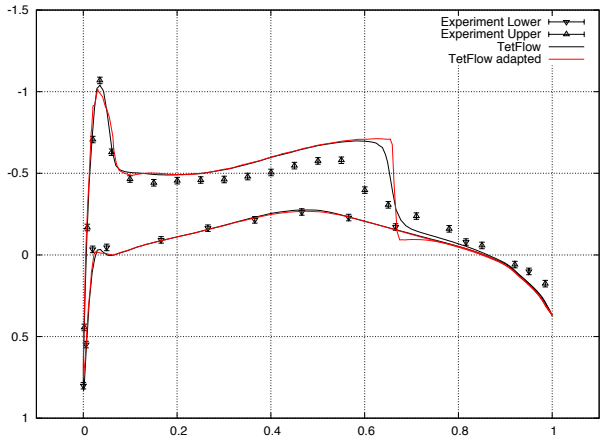

Figure 27. Span station 1 Cp comparisons.

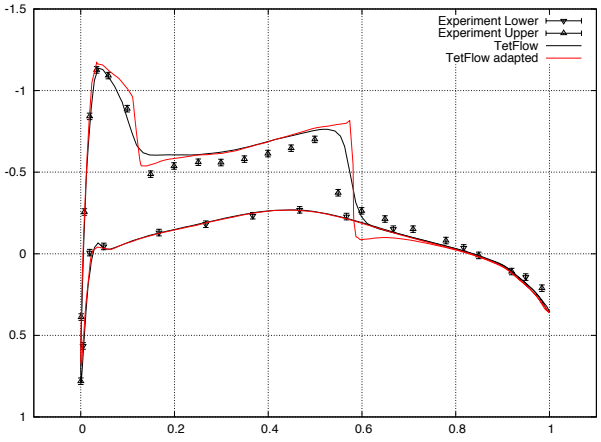

Figure 28. Span station 2 Cp comparisons.

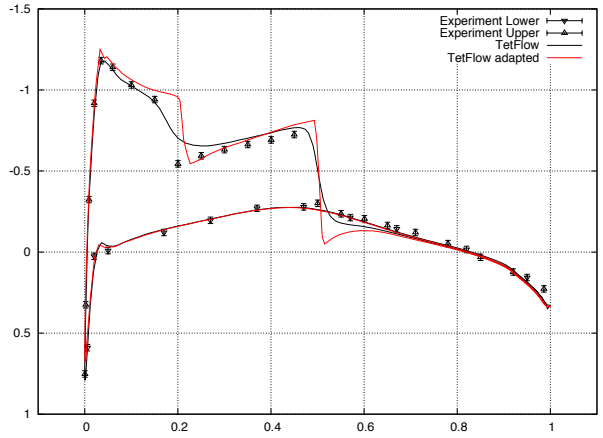

Figure 29. Span station 3 Cp comparisons.

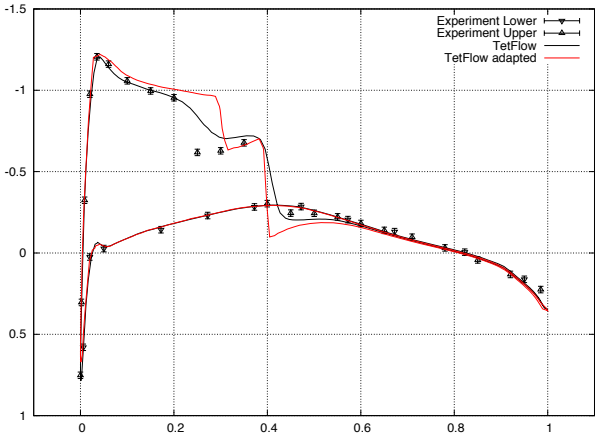

Figure 30. Span station 4 Cp comparisons.

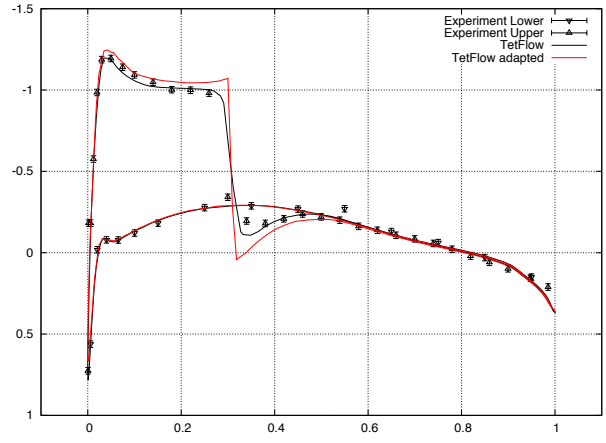

Figure 31. Span station 5 Cp comparisons.

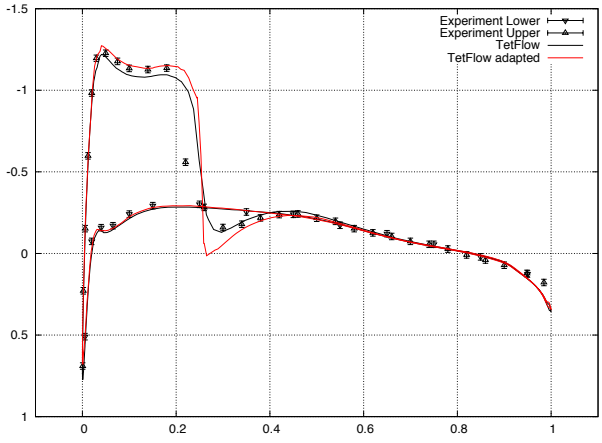

Figure 32. Span station 6 Cp comparisons.

E. Eglin Wing/Pylon/Store

The final case is a popular validation case for moving body problems, the Eglin mutual interference experiment involving a wing with a pylon and finned store.[32] The flow conditions are inviscid Mach 0.95 at zero degrees angle of attack. The finned-store/sting is in the carriage position and is not moving, i.e. the analysis was a steady state, fixed store position. Surface Cp contours are shown in Figure 33. Multiple shocks are evident on the underside of the wing, on the pylon, the store forebody and around the fins. In this figure and the figures that follow the un-adapted mesh is always on the left and the adapted mesh is on the right. The contour color levels, $-1.5 < Cp < 1.0$, are the same for each image. The adaptation function was pressure and the value of the adaptation coefficient was specified as 5000.

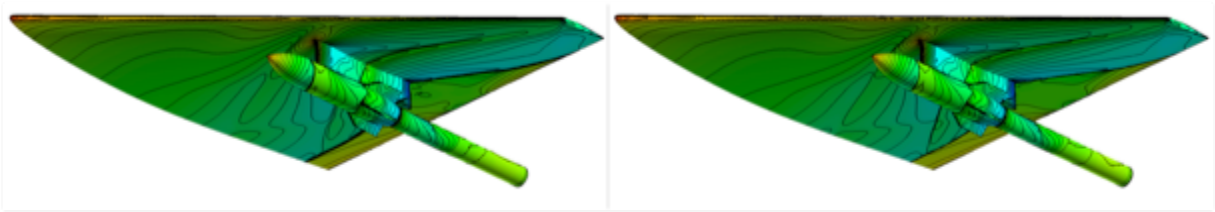

Figure 33. Surface Cp contours for un-adapted mesh on left and adapted mesh on right.

The magnified view, shown in Figure 34, reveals a crisper shock structure in the adapted mesh. This is especially true in the mid-pylon region and the underside of the wing. Away from the shocks the contour lines tend to match between the two solutions. In these regions the gradient of pressure, the adaptation function, is small enough that the influence on the modified relative edge lengths in the condition number weight matrix is minimal.

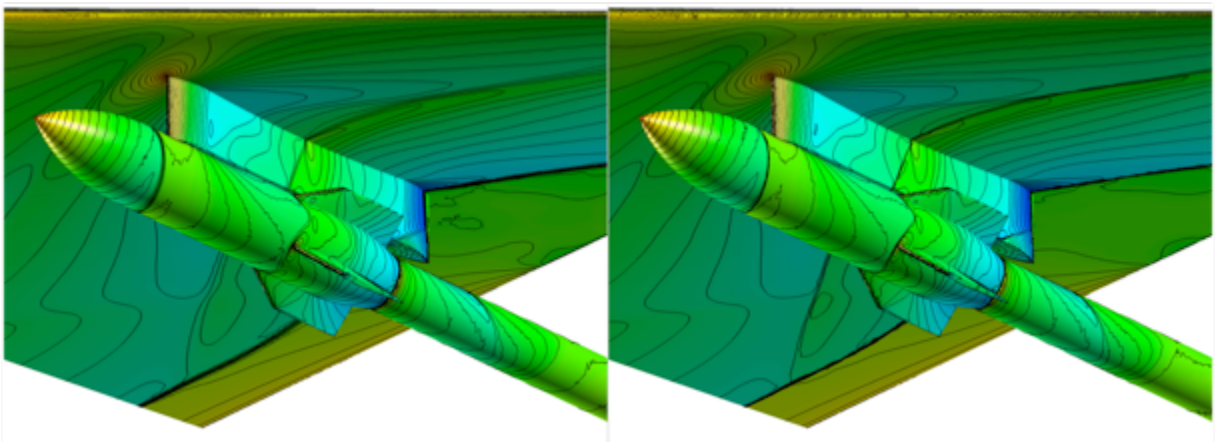

Figure 34. Magnified view of surface Cp contours.

The un-adapted case was smoothed without adapting. The surface meshes are shown in Figure 35. The differences between the smoothed, un-adapted mesh and the original mesh produced by Pointwise are negligible to the eye, so the comparison is not shown. The adapted surface mesh reveals the shock structure vividly. There is even some stretching of the elements ahead of the shocks.

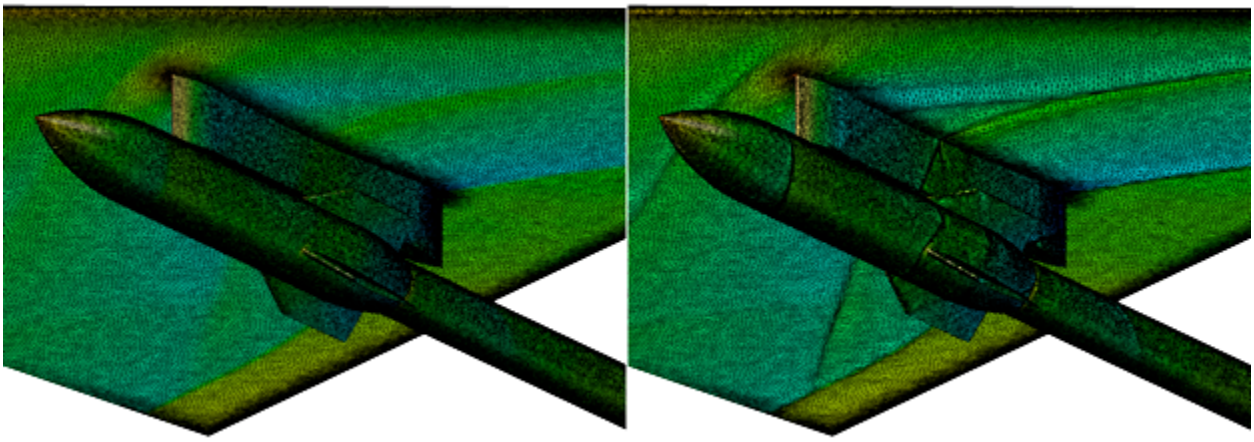

Figure 35. Surface mesh reveals the shocks in the adapted solution.

The shocks on the underside of the wing are displayed as view from below in Figure 36. The shock structure covers the full semi-span of the wing. Two shocks from the outboard side of the pylon coalesce into a single shock about mid-span toward the wing tip. The inboard pylon shock and the shock from the trailing edge of the pylon coalesce into a single shock near the symmetry plane.

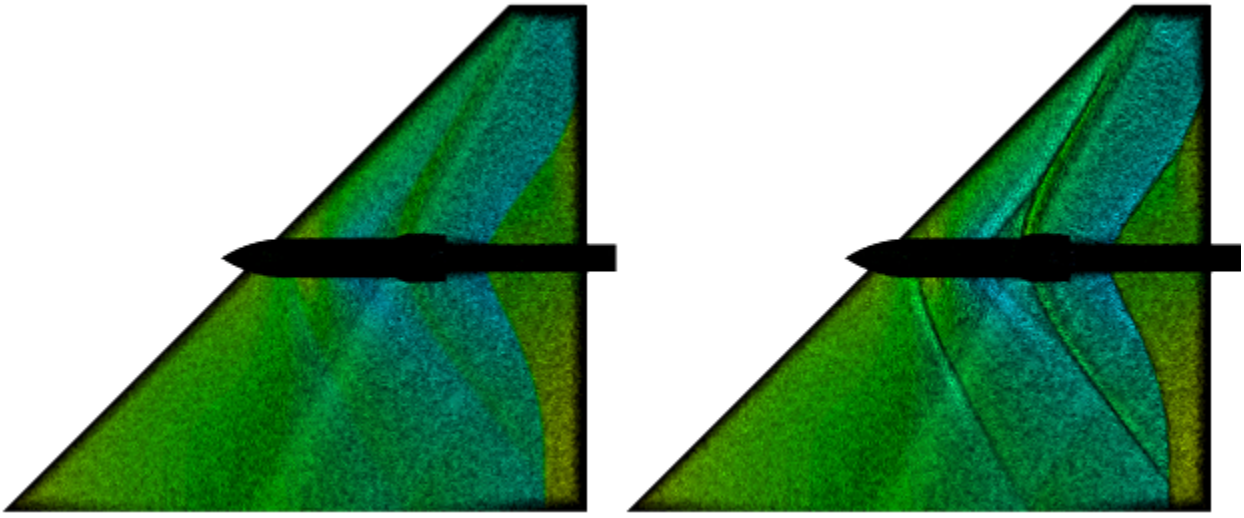

Figure 36. Bottom view of surface meshes on the store and underside of wing.

The magnified bottom view in Figure 37 shows the adaptation of the grid lines at the shock and the stretching of the grid just ahead and behind the shock. This stretching results from the pulling on the mesh from strong pressure gradient at the shock and a milder pressure gradient ahead of the shock, as indicated by the color variations in this figure and the black contour lines in Figure 34.

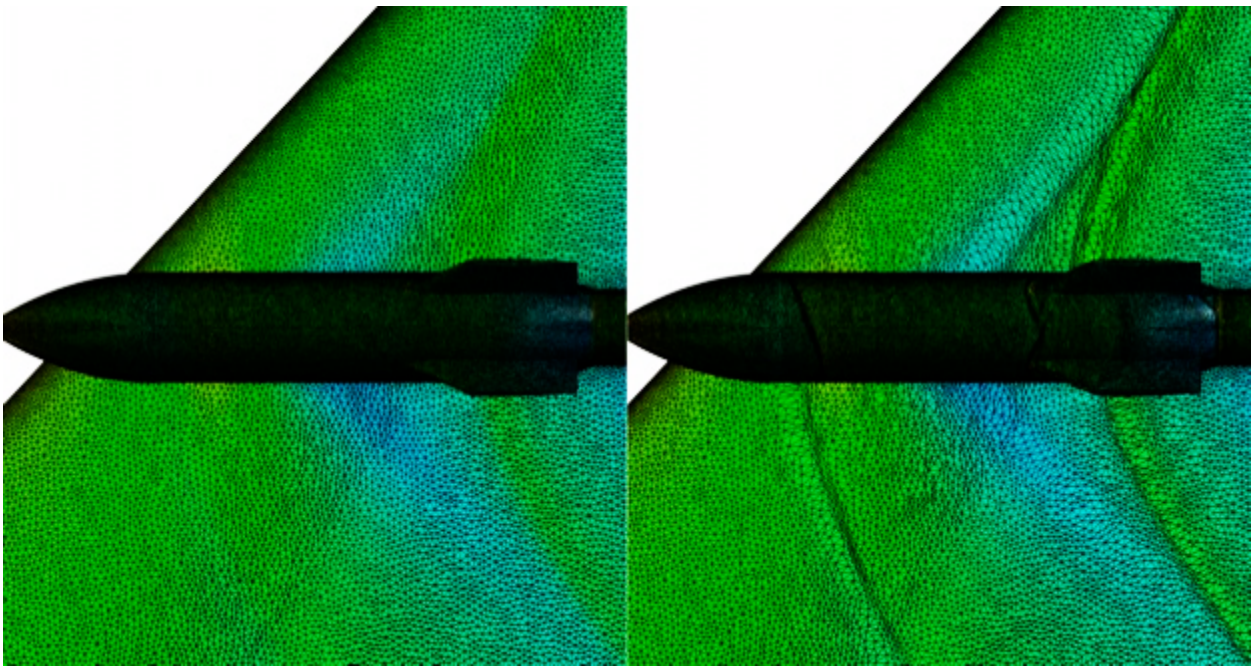

Figure 37. Magnified bottom view of surface meshes.

The mesh on the pylon and store, shown on the right in Figure 38, reveal the complex shock structure. Bow shocks from the fins meet on the store midway between the fins. This shock is transferred to the pylon and reflected off the underside of the wing. Shocks emanate from the fin trailing edges and are re-enforced by the shock from the sting attachment point. This shock is transferred to the pylon and meet with a strong shock near the trailing edge of the pylon.

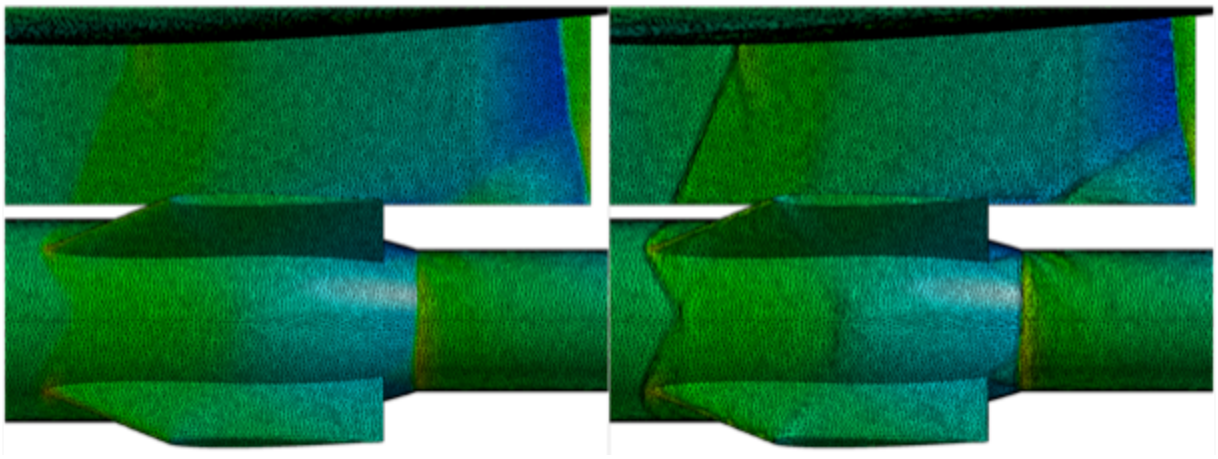

Figure 38. Outboard view of surface mesh on pylon and store.

The mesh movement procedure prevents surface points from crossing boundary edge curves. The store is one boundary. The sting is another. Points shared by the two boundaries are free to move along the boundary curve, but cannot cross over to the other boundary. The same is true for points on the fin/store juncture. Any sharp edge in the mesh must be specified as a boundary curve to ensure the shape of the curve is retained in the smoothed or adapted mesh.

Crinkle cut meshes at the pylon span coordinate are shown in Figure 39 for the pylon leading edge and Figure 40 for the pylon trailing edge. The adaptation in the mesh on the right image is evident between the store forebody and the wing in Figure 39. The strong shock emanating from the store/sting juncture is clearly visible below the sting in Figure 40.

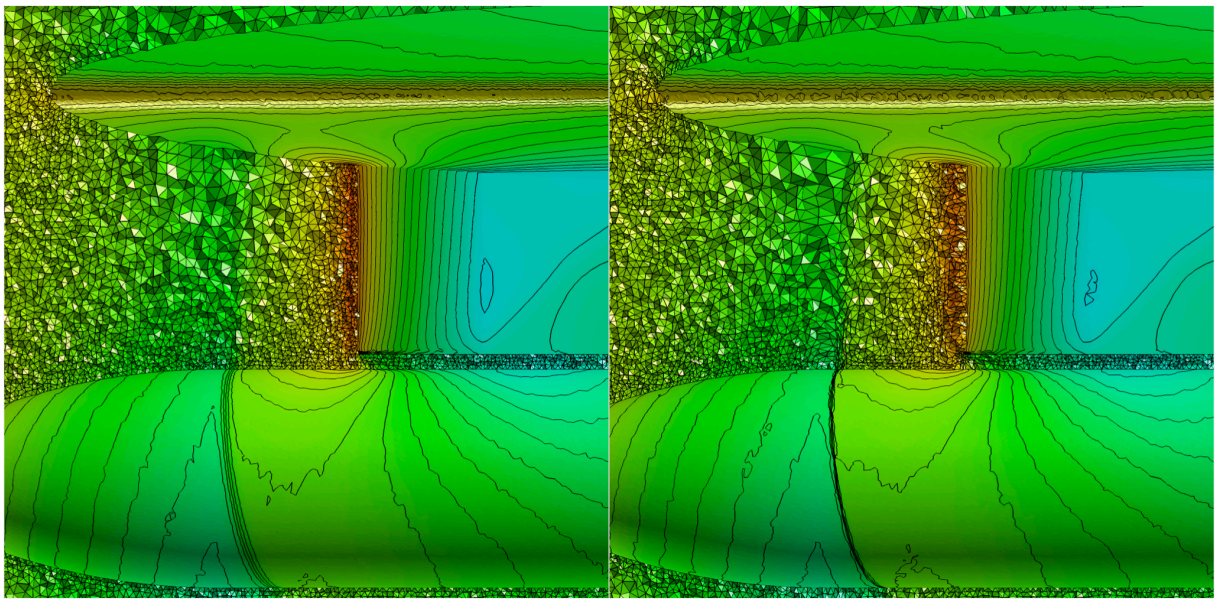

Figure 39. Crinkle cut mesh at pylon span station near the pylon leading edge.

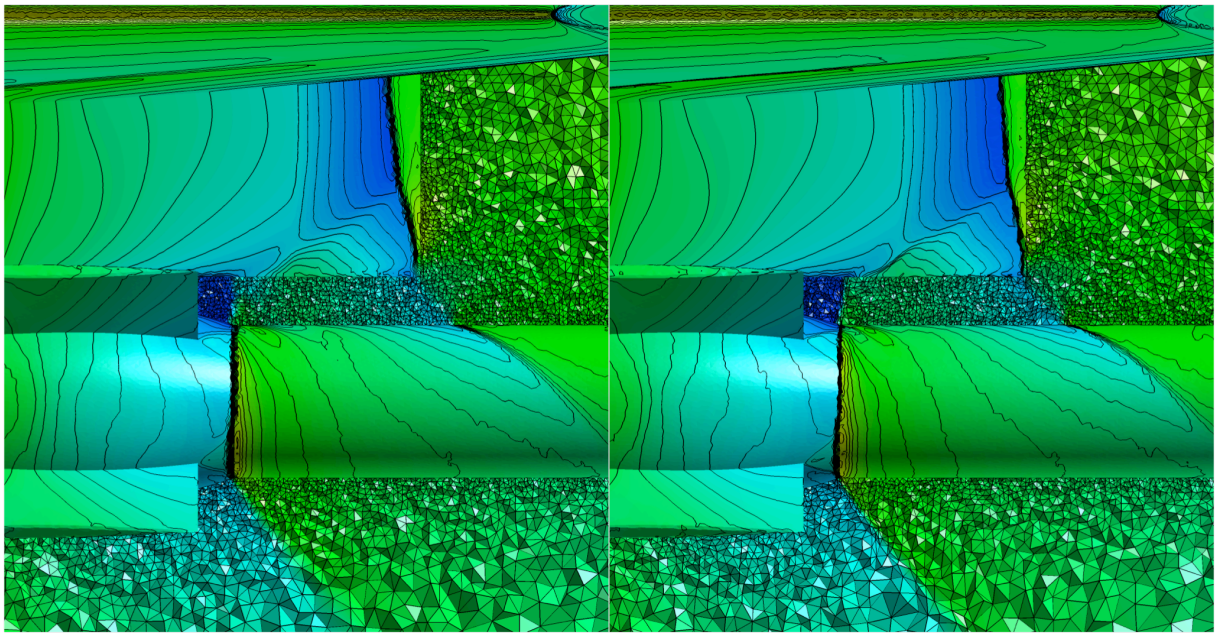

Figure 40. Crinkle cut mesh at pylon span station near the pylon trailing edge.

V. Conclusions

A smoothing method for tetrahedral meshes has been presented that attempts to reposition the mesh node locations to optimize a cost function. The cost function is based on element condition numbers. The basic scheme moves each node in the mesh based on the sensitivity of the element cost function with respect to the coordinates of each corner of the element. Nodal perturbation vectors are constructed from cost-weighted averages of element cost sensitivities. The cost function is comprised of two parts. The basic part is based on the element condition number. Since the condition number is unaware of element inversions the second part is based on the Jacobian of the element. When the element is inverted the Jacobian part will drive the nodes toward positions that produce positive element volumes. Otherwise the condition number form will move the nodes towards positions to optimize element shapes. The condition number component includes a weight matrix that maps the right-angled reference element to a regular tetrahedron. The smoothing scheme attempts to reposition nodes to drive all elements to regular tetrahedra.

Adaptation is achieved by modifying the weight matrix for the reference element shape used to compute the condition number. The relative edge lengths are shortened in the direction of gradients of the user selected adaptation function, such as pressure and Mach number. Valid reference elements are enforced during the process, ensuring valid physical elements in the mesh.

Several example cases are included. All solutions were three-dimensional meshes, though some of the cases are basically two-dimensional solutions. The basic optimization scheme is recovered in the regions where no adaptation function gradients exist. Clustering of the mesh in high gradient regions was demonstrated in all cases.

References

1. Bowyer, A. "Computing Dirichelt Tessellations," *The Computer Journal* Vol. 24, No. 2, 1981, pp. 162-166.
2. Watson, D. F. "Computing the n-dimensional Delaunay tessellation with application to Voronoi polytopes," *The Computer Journal* Vol. 24, No. 2, 1981, pp. 167-172.
3. Lawson, C. L. "Generation of a triangular grid with application to contour plotting." Jet Propulsion Laboratory, Pasadena, CA, 1972.
4. Marcum, D. L., "Generation of Unstructured Grids for Viscous Flow Applications", AIAA-95-0212, 1995.
5. Pirzadeh, S., "Recent Progress in Unstructured Grid Generation", AIAA-92-0445, 30th Aerospace Sciences Meeting, Reno, NV, 1992.
6. Shewchuk, J. R. "Tetrahedral Mesh Generation by Delaunay Refinement," *14th Annual Symposium on Computational Geometry*. Association for Computing Machinery, Minneapolis, MN, 1998, pp. 86-95.
7. Lohner, R., "A Parallel Advancing Front Grid Generation Scheme", AIAA-00-1005, 2000.
8. Degand, C., and Farhat, C. "A Three-Dimensional Torsional Spring Analogy Method for Unstructured Dynamic Meshes," *Computer and Structures* Vol. 80, 2002, pp. 305-316.

9. Knupp, P. M. "Matrix Norms & The Condition Number: A General Framework to Improve Mesh Quality via Node-Movement," *8th International Meshing Roundtable*. South Lake Tahoe, CA, 1999.
10. Freitag, L. A., and Knupp, P. M. "Tetrahedral Element Shape Optimization via the Jacobian Determinant and Condition Number," *8th International Meshing Roundtable*. South Lake Tahoe, CA, 1999.
11. Knupp, P. M. "Winslow Smoothing on Two-Dimensional Unstructured Meshes," *7th International Meshing Roundtable*. Dearborn, MI, 1988.
12. Sahasrabudhe, M., Karman, S., and Anderson, W. K., "Grid Control of Viscous Unstructured Meshes Using Optimization", AIAA-2006-0532, 44th Aerospace Science Meeting and Exhibit, Reno, NV, 2006.
13. Karman, S. L., Anderson, W. K., and Sahasrabudhe, M. "Mesh Generation Using Unstructured Computational Meshes and Elliptic Partial Differential Equation Smoothing," *American Institute of Aeronautics and Astronautics Journal* Vol. 44, No. 6, 2006, pp. 1277-1286.
14. Karman, S. L., "Virtual Control Volumes for Three-Dimensional Unstructured Elliptic Smoohting", AIAA-2011-0895, 49th Aerospace Sciences Meeting and Exhibit, Orlando, FL, 2011.
15. Sahasrabudhe, M. "Unstructured Mesh Generation and Manipulation Based on Elliptic Smoothing and Optimization," *Graduate School of Computational Engineering* Vol. Doctorate of Philosophy Degree, University of Tennessee at Chattanooga, Chattanooga, TN, USA, 2008.
16. Masters, J. S., and Karman, S. L. "Manipulating Boundaries and Viscous Regions of Unstructured Meshes Using Winslow's Equations," *American Institute of Aeronautics and Astronautics Journal* Vol. 50, No. 10, 2012, pp. 2080-2090.
17. Venditti, D. A., and Darmofal, D. L. "Grid Adaptation for Functional Outputs: Application to Two-Dimensional Inviscid Flows," *Journal of Computational Physics* Vol. 176, 2002, pp. 40-69.
18. Park, M. A., "Adjoint-Based, Three-Dimensional Error Prediction and Grid Adaptation", AIAA-2002-3286, 32nd Fluid Dynamics Conference, 2002.
19. Cavallo, P. A., Sinha, N., and Feldman, G. M., "Parallel Unsructured Mesh Adaptation for Transient Moving Body and Aeropropulsive Applications", AIAA-2004-1057, 42nd Aerospace Sciences Meeting and Exhibit, Reno, NV, 2004.
20. Cavallo, P. A., and Grismer, M. J., "Further Extension and Validation of a Parallel Unstructured Mesh Adaptation Package", AIAA-2005-0924, 43rd Aerospace Sciences Meeting and Exhibit, Reno, NV, 2005.
21. Jones, W. T., Nielsen, E. J., and Park, M. A., "Validation of 3D Adjoint Based Error Estimation and Mesh Adaptation for Sonic Boom Prediction", AIAA-2006-1150, 44th Aerospace Sciences Meeting and Exhibit, Reno, NV, 2006.
22. Anderson, D. A. "Equidistribution Schemes, Poisson Generators, and Adaptive Grids," *Applied Mathematics and Computation* Vol. 24, 1987, pp. 211-227.
23. Anderson, D., "An Adaptive Grid Scheme Controlling Cell Area/Volume", AIAA-87-0202, 1987.
24. "Pointwise." Pointwise, Inc., http://www.pointwise.com.
25. "FieldView." Intelligent Light Inc., http://www.ilight.com.

26. Anderson, W. K., Newman, J. C., Whitfield, D. L., and Nielsen, E. J. "Sensitivity Analysis for the Navier-Stokes Equations on Unstructured Meshes Using Complex Variables," *AIAA Journal* Vol. 39, No. 1, 2001, pp. 56-63.
27. Aubert, P., Cesare, N. D., and Pironneau, O. "Automatic differentiation in C++ using expression templates and application to a flow control problem," *Computing and Visualization in Science* Vol. 3, 2001, pp. 197-208.
28. Phipps, E., and Pawlowski, R. "Efficient Expression Templates for Operator Overloading-based Automatic Differentiation." Cornell University Library, 2012.
29. Biedron, R. T., and Thomas, J. L., "Recent Enhancements To The FUN3D Flow Solver For Moving-Mesh Applications", AIAA-2009-1360, 2009.
30. Schmitt, V., Charpin, F. "Pressure Distributions on the ONERA-M6-Wing at Transonic Mach Numbers," *Experimental Data Base for Computer Program Assessment*. AGARD AR 138 ed., AGARD Fluid Dynamics Panel Working Group 04, 1979.
31. NASA. "ONERA M6 Wing," *NPARC Alliance Validation Archive*. http://www.grc.nasa.gov/WWW/wind/valid/m6wing/m6wing.html.
32. Heim, E. R. "CFD Wing/Pylon/Finned Store Mutual Interference Wind Tunnel Experiment." Calspan Corporation/ AEDC Operations, Arnold Engineering Development Center, 1991.