Geometrically Flexible and Efficient Flow Analysis of High Speed Vehicles Via Domain Decomposition, Part 1, Unstructured-grid Solver for High Speed Flows

Jeffery A. White, NASA Langley Research Center
Robert A. Baurle, NASA Langley Research Center
Bradley J. Passe, Analytical Mechanics Associates (Currently Lockheed Martin)
Seth C. Spiegel, NASA Glenn Research Center
Hiroaki Nishikawa, National Institute of Aerospace

The ability to solve the equations governing the hypersonic turbulent flow of a real gas on unstructured grids using a spatially-elliptic, 2nd-order accurate, cell-centered, finite-volume method has been recently implemented in the VULCAN-CFD code. This paper describes the key numerical methods and techniques that were found to be required to robustly obtain accurate solutions to hypersonic flows on non-hex-dominant unstructured grids. The methods and techniques described include: an augmented stencil, weighted linear least squares, cell-average gradient method, a robust multidimensional cell-average gradient-limiter process that is consistent with the augmented stencil of the cell-average gradient method and a cell-face gradient method that contains a cell skewness sensitive damping term derived using hyperbolic diffusion based concepts. A data-parallel matrix-based symmetric Gauss-Seidel point-implicit scheme, used to solve the governing equations, is described and shown to be more robust and efficient than a matrix-free alternative. In addition, a y^+ adaptive turbulent wall boundary condition methodology is presented. This boundary condition methodology is deigned to automatically switch between a solve-to-the-wall and a wall-matching-function boundary condition based on the local y^+ of the 1st cell center off the wall. The aforementioned methods and techniques are then applied to a series of hypersonic and supersonic turbulent flat plate unit tests to examine the efficiency, robustness and convergence behavior of the implicit scheme and to determine the ability of the solve-to-the-wall and y^+ adaptive turbulent wall boundary conditions to reproduce the turbulent law-of-the-wall. Finally, the thermally perfect, chemically frozen, Mach 7.8 turbulent flow of air through a scramjet flowpath is computed and compared with experimental data to demonstrate the robustness, accuracy and convergence behavior of the unstructured-grid solver for a realistic 3-D geometry on a non-hex-dominant grid.

Introduction

The use of computational fluid dynamics (CFD) to characterize the external and internal flows typical of hypersonic vehicles is extremely challenging due to the complex physical modeling required to compute these flows. Nonetheless, over the past two decades, multiple CFD codes have been developed that are capable of computing these types of flows [1,2,3,4]. With the notable exception of the VULCAN-CFD code, the codes developed have almost exclusively employed unstructured grid methodologies. For the most part, these unstructured-grid codes provide significantly improved geometric flexibility at the expense of increased computational overhead, usually in the form of an increase in the number of processors required, relative to structured-grid codes. To address this additional overhead, there has been a concerted effort by the CFD community at large to develop unstructured grid codes that scale to "many" thousands of processors so as to either enable computation of "Grand Challenge Problems" or to perform less complex engineering analyses rapidly enough that they are relevant to engineering design time scales. Unfortunately, most engineers still work in a computational environment having finite resources where many programs compete for computational access. This competition naturally creates pressure on resource managers to configure their batch queuing software such that the time spent "in the queue" for jobs requiring "many thousands" of processors can become untenable from an engineering design point of view. This problem is further exacerbated in restricted access computational environments because computational resources are usually severely limited by the nature of the work. Moreover, as the number of processors required to rapidly compute a single "design point" solution increases, the number of processors available to compute other points in the design space decreases linearly, thereby adversely affecting the time required to cover the design space.

Historically, the development strategy for the VULCAN-CFD code has been to develop and implement solution methodologies that are efficient when computing the flows of interest to the scramjet community. This strategy resulted in the development of a "multi-region" framework in VULCAN-CFD [5,6] wherein the user has the ability to decompose the computational domain into multiple spatially-elliptic flow and/or parabolic/hyperbolic flow subdomains or "regions" where the flow solution is computed using the algorithm most appropriate for the flow physics of each region. To date, this multi-region framework, has been instantiated by solving the spatially-elliptic flow regions with a structured-grid implicit time marching scheme and the parabolic/hyperbolic flow regions with a structured-grid implicit space marching scheme . The issue of geometric complexity has been addressed via the use of multi-block curvilinear structured grids within each region. However, when geometric complexity becomes too extreme, the time required to

generate the structured multi-block curvilinear structured grids can become prohibitive. However, given the aforementioned computational resource constraints and the maturation of unstructured-grid flow solution technology, the incorporation an unstructured-grid spatially-elliptic flow solver capability into the VULCAN-CFD multi-region domain decomposition framework is desirable. Therefore, work to accomplish this goal was initiated utilizing a code developed as part of a hybrid structured/unstructured grid NASA Research Award (NRA), funded by the Fundamental Aeronautics Program, recently completed by North Carolina State University as described by Spiegel et al. [7,8].

The original intent of this paper was to describe and explore the use of this unstructured-grid spatially-elliptic flow solver within the multi-region VULCAN-CFD framework. However, when state-of-the-art non-hex dominant unstructured grids were generated for realistic hypersonic vehicle geometries and simulated at "representative" hypersonic flow conditions, numerical accuracy and robustness deficiencies as well as parallel scaling challenges were exposed. Therefore, incorporation of the unstructured-grid spatially-elliptic solver into the multi-region framework was temporarily deferred to allow the identified deficiencies to be addressed. These deficiencies were addressed through the implementation of the best practices available in the literature for 2nd-order, cell-centered, finite-volume, unstructured-grid flow solvers, namely; 1) constructing cell-average gradients by using a weighted expanded stencil linear least squares method [9-11], 2) reconstructing the state variables to the cell faces by using the UMUSCL higher-order extrapolation method of Burg [12], 3) limiting the cell-average gradients used in the higher-order reconstruction by using a multidimensional limiting processes that satisfies a multidimensional maximum principle [13], and 4) constructing cell face gradients for the viscous fluxes by using face tangent schemes [11,14,15] as well as a cell-face gradient construction scheme derived from analysis of a hyperbolic relaxation-system model for diffusion [16].

To improve convergence of the solver to steady state, the implicit scheme was rewritten to improve the left hand side as compared to the approximations used by the original LU-SGS and matrix-free SGS schemes [7,11,17,18] and to couple the partitions during the linear solve subiterative process. These improvements, were realized by developing an automatically differentiated 1st-order inviscid Jacobian (via modern FORTRAN's operator overloading capability), hand differentiated thin-layer viscous Jacobian and implicit, automatically differentiated, boundary conditions. In addition, inter-partition communication during the linear solve was added to improve coupling of the partitions. Parallel scaling issues were addressed by rewriting/refactoring all code that did not scale well. Furthermore, a novel y^+ adaptive turbulent wall boundary condition approach was also developed and implemented. This approach, allows the turbulent wall boundary condition algorithm to choose between using, a solve-to-the-wall or a wall-matching-function wall boundary condition, for each wall cell face based on the local y^+_{wall}. Finally, all thermodynamic, chemical kinetic and turbulence models, as well as all relevant boundary conditions available in the structured-grid solver, were implemented in the unstructured-grid solver. Given the scope and magnitude of these changes, the current paper focuses on describing, the aforementioned numerical methods, algorithms, and techniques that were used to discretize and solve the governing equations for a general turbulent thermo-chemical non-equilibrium flow on realistic non-hex dominant unstructured grids. The resulting code is then tested for simple high speed turbulent flow unit problems as well as a realistic geometry at flow conditions that are relevant to air-breathing high speed vehicles [19].

Unstructured-Grid Solver Enhancements

Cell-Average Gradient Construction:

Cell-average gradients are perhaps the most important and one of the most difficult quantities to obtain accurately and robustly on irregular, unstructured grids. The cell-average gradients are required to accomplish three things when computing the residual of the discrete equations for each time step/cycle of the solution process. These are: 1) to perform the higher-order reconstruction when computing the inviscid fluxes, 2) to compute the cell-face gradient when computing the viscous fluxes, and 3) to compute the source terms for the turbulence modeling transport equations. Moreover, there is evidence in the literature that a different definition of the cell-average gradient may be required to compute each of these quantities [11]. The NRA-developed code used a Green-Gauss method to compute the cell-average gradient of a scalar, q for each cell i, $\overline{\nabla q}_i$. and used this cell-average gradient for the three purposes described above. For a 2nd-order accurate, cell-centered scheme, with midpoint quadrature, the Green-Gauss method has the following discrete form

$$\overline{\nabla q}_i = \frac{1}{V_i} \sum_{f=1}^{N_f} \overline{q}_f \hat{n}_f A_f . \quad (1)$$

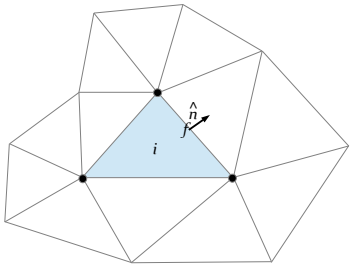

Figure 1: Green-Gauss cell-average gradient terminology.

Where f is the cell-face midpoint, V_i is the cell volume, $\bar{q}_f$ is the average of q over face f, N_f is the number of faces, and $\hat{n}_f$ and A_f are the cell-face unit normal vector and cell-face area, respectively, as illustrated in 2-D on a triangular grid in figure 1.

If $\bar{q}_f$ is known analytically then this form is linear exact. However, significant errors can be introduced when computing $\bar{q}_f$. Depending on the manner in which $\bar{q}_f$ is computed, the resulting approximation of $\overline{\nabla q_i}$ can be consistent and 2nd-order accurate or 0th-order accurate and therefore inconsistent. In the NRA supplied code the method used to compute $\bar{q}_f$, was an inverse distance weighted average of the face neighbors of the cell-average variables, q_i and q_j, to the cell face midpoint, f, of the form

$$\bar{q}_f = \frac{\left(d_{jf}\, q_i + d_{if}\, q_j\right)}{\left(d_{jf} + d_{if}\right)} \quad (2)$$

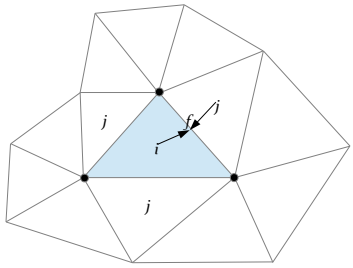

Figure 2: A candidate cell face average stencil for computing a Green-Gauss gradient or weighted linear least squares.

where $d_{if} = \left|\vec{r}_f - \vec{r}_i\right|$ is the magnitude of the vector drawn from cell center i to cell face midpoint (see figure 2), $\vec{d}_{if}$. Unfortunately, the literature clearly demonstrates that this approach to the computation of $\bar{q}_f$ results in a Green-Gauss gradient that is both inconsistent and 0th-order accurate except on completely regular grids where $\vec{d}_{if}$ and $\vec{d}_{jf}$ are colinear [10].

While no cell-average gradient method has been found to be accurate for all arbitrary polygons, with some caveats [10], the weighted linear least squares method has been found to be the preferred method when computing cell-average gradients [9,10] for both node-centered and cell-centered schemes. Therefore, based on the results in the literature [9,10,11], the weighted linear least squares method was chosen to replace the Green-Gauss method. A 3-D weighted linear least squares method can be constructed to compute a cell-average gradient of a scalar, q_i, for cell i given a stencil consisting of some combination of the surrounding cells, $j=1$ through $N_{jstencil}$, The smallest possible 2-D stencil is shown in figure 2. A least squares fit for the cell-average gradient can be written as

$$\overline{\nabla q_i} = \left\{\sum_{k=1}^{3}\left(\frac{\partial q}{\partial x_k}\right)\hat{l}_k\right\}_i = \left\{\sum_{k=1}^{3}\left[\sum_{j=1}^{N_{jstencil}} C_{j,k}\left(q_j - q_i\right)\right]\hat{l}_k\right\}_i \quad (3)$$

where $C_{j,k}$ are the least squares coefficients for each cell of the stencil and $\hat{l}_k$ are the components of the Cartesian unit vector. The coefficients for an inverse distance weighted least squares cell-average gradient are computed for each cell in the stencil with the expression

$$C_{j,k} = \sum_{m=1}^{3}\left\{L^{-1}_{k,m}\left[\left(d_{ij}\right)^{\frac{1}{\omega_{ng}}}\right]^2 \vec{d}_{ij,m}\right\} \quad (4)$$

where $L_{k,m}$ are elements of the *L* matrix given by

$$L_{k,m}=\sum_{j=1}^{N_j}\left[\left(d_{ij}\right)^{\frac{1}{\omega_{ng}}}\right]^2\vec{d}_{ij,k}\vec{d}_{ij,m} \quad (5)$$

and $\frac{1}{\omega_{ng}}$ is the inverse distance weighting coefficient, $\vec{d}_{ij}=\vec{r}_j-\vec{r}_i$ is the Cartesian vector drawn from cell center *i* to cell center *j*, $\vec{d}_{ij,k}$ and $\vec{d}_{ij,m}$ are the Cartesian components of $\vec{d}_{ij}$ and $d_{ij}=\left|\vec{r}_j-\vec{r}_i\right|$. It is important to note that examination of equations 4 and 5 reveal that the coefficients are only a function of the number and location of the stencil members. Therefore, if the grid is static, i.e., the grid point coordinates and their connectivity do not change with respect to each residual evaluation, then $C_{j,k}$ may be precomputed and stored. Furthermore, in the spirit of [11], the weighted linear least squares method implemented allows one to define *ng*<=3 inverse weight coefficients for use by the inviscid flux, viscous flux and turbulence source terms. Currently, based on numerical experimentation, *ng*=2 is recommended, with inverse weight coefficients of 0.0, for the inviscid flux, and -1.0 for the viscous flux, and turbulence source terms. However, the number and size of these coefficients is subject to revision as more experience is gained using the code over a broader range of flows.

The stencil of the linear least squares average gradient operator must have at least 3 participating cells in 2-D and at least 4 cells in 3-D to be well posed. This condition can be met using a face neighbor cell stencil, *fn1* as illustrated in the 2-D example shown in figure 3. However, on highly skewed grids the *fn1* stencil may become biased and give rise to instabilities [9-11]. Fortunately these instabilities can be alleviated by augmenting the stencil to reduce or eliminate the bias. Two approaches have been considered to address this problem. The first approach is to augment the *fn1* stencil cells with all of the cells that share a face with the cells of the *fn1* stencil, resulting in a face neighbors of face neighbors stencil, *fn2*. The second approach is to augment the *fn1* stencil with all the cells that share a node with the nodes of cell *i*, resulting in the node neighbors stencil, *nn* shown in figure 3. To date, only the *fn2* stencil has been implemented due to it being the most straightforward to accomplish using existing data structures in the code. However, as illustrated in figure 3, the *fn*2 stencil can result in gaps in the stencil that do not exist in the *nn* stencil. Furthermore, note that in 2-D on triangular grids, the *fn2* stencil is a subset of the *nn* stencil, while the converse is true when on quadrilateral grids, i.e., the *nn* stencil is a subset of the *fn2* stencil. Also note that figure 4 illustrates that the *nn* stencil is more spatially compact than the *fn2* stencil for hexahedral grids, and by analogy, prismatic grids. In 3-D, the *nn* stencil on hexahedral and prismatic grids is also not a subset of the *fn2* stencil. The downside of using the *nn* stencil in 3-D is that there will usually be more cells in the *nn* stencil than in the *fn2* stencil thereby requiring more storage and operations to compute the gradient. However, due to the possible existence of gaps in the *fn2* stencil and the 3-D spatial compactness of the *nn* stencil, the *nn* stencil will be implemented in the code in the near future.

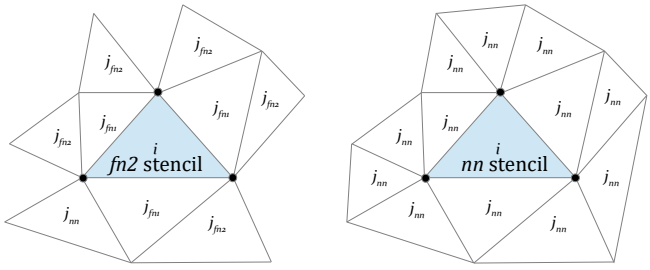

Figure 3: The fn2 and nn least squares stencils for computing the cell-average gradient on a triangular grid.

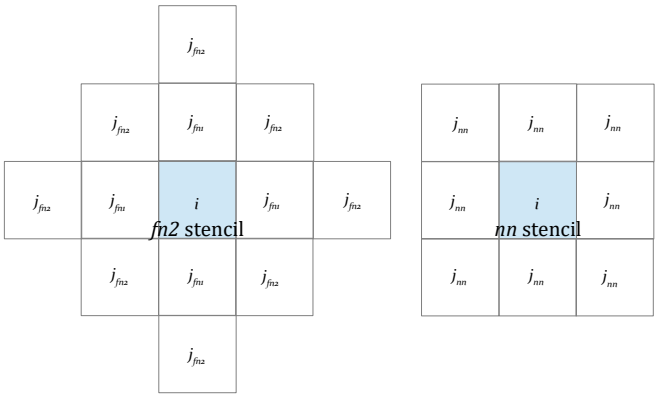

Figure 4: The fn2 and nn least squares stencils for computing the cell-average gradient on a quadrilateral grid.

Inviscid Flux Cell Face State Variable Reconstruction:

The inviscid fluxes in the unstructured-grid solver are computed using an upwind flux scheme. Currently either the LDFSS [21] or the HLLC scheme [22] can be selected. Both of these schemes require that the primitive variables, q, be specified on the left (L) and right (R) sides of the cell face midpoint, f, as shown in figure 5. The primitive variables are

$$q = \left(\frac{\rho_1}{\rho}, \ldots, \frac{\rho_{ncs}}{\rho}, \rho, u, v, w, P, k, \omega\right) \quad \text{for thermal equilibrium, or}$$

$$q = \left(\frac{\rho_1}{\rho}, \ldots, \frac{\rho_{ncs}}{\rho}, \rho, u, v, w, e_{ve}, P, k, \omega\right) \quad \text{for thermal nonequilibrium}$$

A 1st-order accurate scheme results if the cell-average values to the left, (i) and right, (j) are used. A higher-order accurate scheme results when the L and R primitive variables are reconstructed to the cell face midpoint with an extrapolation or interpolation method based on the left and right cell-average primitive variables and gradients as given by

$$q_f^L = q_i + \overline{\nabla q}_i \cdot \vec{r}_{if} \quad (6)$$

$$q_f^R = q_j + \overline{\nabla q}_j \cdot \vec{r}_{jf}. \quad (7)$$

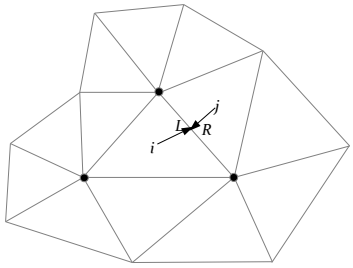

Figure 5: Higher-order reconstruction of the L and R states to the cell face midpoint.

In addition to the scheme above, which is an unstructured-grid interpretation of Fromm's scheme [23], the higher-order variable extrapolation (or UMUSCL) reconstruction scheme [8], was also implemented to reduce the dissipation of the scheme further. The UMUSCL scheme can be written as

$$q_f^L = q_i + \frac{\chi}{2}(q_j - q_i) + (1-\chi)\overline{\nabla q}_i \cdot \vec{r}_{if} \quad (8)$$

$$q_f^R = q_j + \frac{\chi}{2}(q_i - q_j) + (1-\chi)\overline{\nabla q}_j \cdot \vec{r}_{jf} \quad (9)$$

where χ is used to control the behavior and the 1-D order of accuracy of the scheme when the flow is smooth.

1. χ = 0, gives Fromm's scheme
2. χ = -1, gives a 2nd-order fully upwind MUSCL-type scheme
3. χ = 1/3, gives a 3rd-order upwind biased MUSCL-type scheme

Inviscid Flux Cell-Average Gradient Limiter Construction:

When computing high speed flow, discontinuities will usually exist somewhere in the computational domain. However, in the vicinity of these discontinuities, the higher-order reconstruction of the state variables to the cell face used to achieve 2nd-order accuracy of the inviscid flux scheme will produce oscillations in the flow solution that will eventually cause the computation to fail. These oscillations can be suppressed by locally forcing the reconstruction to be 1st-order through the use of some sort of gradient limiter. A gradient limiter can be implemented in two different ways for the UMUSCL scheme, with a 1-D "face" based limiter approach or a multidimensional "stencil" based limiter approach. A face-based limiter for the UMUSCL scheme can be written as

$$\widetilde{q_f^L} = q_i + \chi\widetilde{\beta^L} + (1-\chi)\widetilde{\alpha^L} \quad (10)$$

$$\widetilde{q_f^R} = q_j + \chi\beta^R + (1-\chi)\alpha^R \quad (11)$$

where the limited left and right gradients, $\widetilde{\alpha^{L,R}}$ and $\widetilde{\beta^{L,R}}$ are

$$\widetilde{\alpha^{L,R}} = F\left(\alpha^{L,R}, \beta^{L,R}\right)_{(limiter)} \quad \text{and} \quad \widetilde{\beta^{L,R}} = F\left(\beta^{L,R}, \alpha^{L,R}\right)_{(limiter)}$$

the unlimited left and right 1-D gradients $\alpha^{L,R}$ and $\beta^{L,R}$ are

$$\alpha^{L} = \overline{\nabla q_i} \cdot \vec{r}_{if} \quad \text{and} \quad \beta^{L} = \frac{1}{2}\left(q_j - q_i\right) \quad (12)$$

$$\alpha^{R} = \overline{\nabla q_j} \cdot \vec{r}_{jf} \quad \text{and} \quad \beta^{R} = \frac{1}{2}\left(q_i - q_j\right) \quad (13)$$

and, in the case of the van Leer limiter [24]

$$F\left(\Delta_1, \Delta_2\right)_{(van\,Leer)} = \frac{\left(\Delta_2 \left|\left(\Delta_1\right)\right| + \Delta_1 \left|\left(\Delta_2\right)\right|\right)}{\left(\left|\left(\Delta_1\right)\right| + \left|\left(\Delta_2\right)\right| + \epsilon\right)} \quad (14)$$

where ϵ is on the order of 1.0×10^{-12}. While the face-based scheme has been found to be reasonably effective on smooth hexahedral grids, its effectiveness deteriorates on truly unstructured grids. Therefore, two stencil-based limiters were implemented. The first method is a generalization of the approach used to form stencil-based gradient limiters by Barth and Jesperson [25] and later by Venkatakrishnan [26]. The second method is the multidimensional limiter process (MLP) of Park and Kim [13]. Both of these approaches are referred to as stencil-based limiters herein because they use information from all of the cells that make up the stencil that was used to compute the cell-average gradient. Figure 6 presents the cells involved in the *fn2* stencils used to compute the limiter coefficients in cells *i* and *j*, where the cells labeled i_s and j_s only participate in the *i* and *j* cell stencils, respectively, and cells labeled i_s,j_s are cells that participate in both cell stencils. These stencil-based limiter approaches compute cell-limiter coefficients that are used to

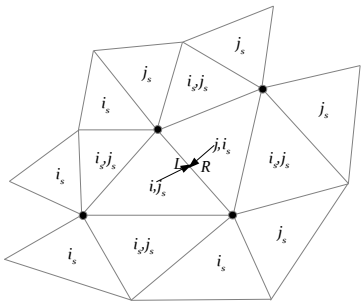

Figure 6: The cells that participate in the construction of the fn2 stencil-based limiter coefficients used to reconstruct data to the face shared by cells i and j.

limit the higher-order reconstruction that, when applied to the UMUSCL higher-order reconstruction scheme, results in equations for the left and right states having the form

$$\widetilde{q_f^L} = q_i + \Phi_i(q_i)\left[\frac{\chi}{2}\left(q_j - q_i\right) + \left(1-\chi\right)\overline{\nabla q_i} \cdot \vec{r}_{if}\right] \quad (15)$$

$$\widetilde{q_f^R} = q_j + \Phi_j(q_j)\left[\frac{\chi}{2}\left(q_i - q_j\right) + \left(1-\chi\right)\overline{\nabla q_j} \cdot \vec{r}_{jf}\right] \quad (16)$$

where $\Phi_i(q_i)$ and $\Phi_j(q_j)$ are the cell-limiter coefficients that are used to limit the reconstruction consistently for all faces of the cells *i* and *j*, respectively. Figure 7 presents the *fn2* stencil cells that participate in the computation of the cell

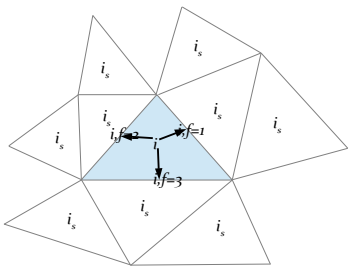

Figure 7: The fn2 stencil cells and face midpoints that participate in the construction of the stencil-based limiter coefficients for cell i where $N_{i,faces} = 3$.

i limiter coefficient, $\Phi_i(q_i)$. The methods of [25] and [26] compute the cell limiter coefficients for each cell, *i*, using

$$\Phi_i(q_i^{fn2}) = min\left(1,\ \begin{cases} \phi_f\left(\dfrac{q_i^{max(fn2)} - q_i}{q_f - q_i}\right), if\ (q_f - q_i) > 0 \\ \phi_f\left(\dfrac{q_i^{min(fn2)} - q_i}{q_f - q_i}\right), if\ (q_f - q_i) < 0 \\ 1 \qquad\qquad\qquad if\ (q_f - q_i) = 0 \end{cases}\right), f = 1 \rightarrow N_{i,faces} \quad (17)$$

where $q_i^{max(fn2)}$ and $q_i^{min(fn2)}$ are the maximum and minimum values of *q* of the *fn2* stencil cells respectively and q_f is computed at each cell face midpoint by using the unlimited form of UMUSCL

$$q_f = q_i + \frac{\chi}{2}(q_j - q_i) + (1-\chi)\overline{\nabla q_i} \cdot \vec{r}_{if} \quad (18)$$

where the value of χ is consistent with the value used in equation 15. The face-limiter coefficient, ϕ_f, is computed using a generalization of the form used in [25] and [27]

$$\phi_f\left(\frac{b}{a}\right) = \frac{F\left(a, \frac{b}{2}\right)_{(limiter)}}{a} \quad (19)$$

where, $F\left(a, \frac{b}{2}\right)_{(limiter)}$, can be any limiter function found in the literature. For example, [25] used the minmod function whereas [26] used a modified form of the van Albada function [27]. Currently VULCAN-CFD allows the use of the Sweby [30], van Leer [21], van Albada, Venkatakrishnan [26], and Koren [29] limiter functions consistent with the structured-grid solver. However, while we have found that this stencil-based limiter approach improves the shock capturing capability of the code significantly compared to the face-based limiter approach, some oscillations can still occur in the vicinity of the very strong shocks, such as found near blunt bodies. For these flows, $\Phi_i(q_i^{fn2})$ requires further augmentation via a heuristic pressure limiter such as proposed by Gnoffo [30].

Adding heuristic limiters, such as the aforementioned pressure limiter, has the potential to adversely affect the flow solution by adding too much dissipation where it is not needed, e.g., in shock boundary layer flows where the physical viscosity should prevent oscillations from occurring such that the pressure limiter is no longer needed. Therefore, as mentioned previously, the multidimensional limiter procedure (MLP) of Park and Kim [13] has also been implemented in an attempt to further improve the discontinuity capturing capability and robustness of the unstructured-grid solver near strong shocks, while reducing limiting and thus dissipation in the vicinity of shock boundary layer interactions. The MLP limiter is a stencil-based limiter that also attempts to "define and implement monotinicity in multi-dimensions" by strictly enforcing the maximum principle. Park and Kim state that the central premise of MLP is to "control the distribution of both cell center and cell node physical properties to mimic the multidimensional nature of the flow physics." They state that this can be accomplished based on the observation that a well controlled reconstruction of the cell-centered solution to the nodes can be used to construct a limiting process that is both multidimensional and monotone. For a detailed discussion of the mathematical proof of this concept see [12].

The implementation of the MLP approach proceeds in a manner similar to that of the previously described stencil-based limiting approach with the key difference being that the construction of the cell limiter coefficient uses a reconstruction of the solution to the cell nodes instead of to the cell face midpoints. Figure 8 illustrates the *nn* stencil cells and nodes that participate in the computation of the cell *i*, MLP_{nn} "node-based" limiter coefficient, $\Phi_i(q_i^{MLP_{nn}})$.

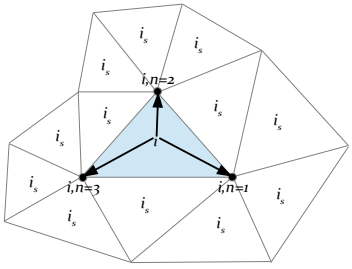

Figure 8: The nn stencil cells and nodes that participate in the construction of the MLP$_{nn}$ limiter coefficients for cell i where $N_{i,nodes}$ = 3.

This results in equation 17 being split into two equations

$$\Phi_i(q_i^{MLP_{nn}}) = min(1, [\Psi_{i,n}^{nn}(q_{i,n}), n=1 \rightarrow N_i, nodes]) \quad (20)$$

and

$$[\Psi_{i,n}^{nn}(q_{i,n}) = min(1, \begin{cases} \phi_{i,n}(\frac{q_{i,n}^{max(ncn)} - q_i}{q_{i,n} - q_i}), if\,(q_{i,n} - q_i) > 0 \\ \phi_{i,n}(\frac{q_{i,n}^{min(ncn)} - q_i}{q_{i,n} - q_i}), if\,(q_{i,n} - q_i) < 0 \\ 1 \qquad\qquad\qquad if\,(q_{i,n} - q_i) = 0 \end{cases}), n=1 \rightarrow N_{i,nodes}, \quad (21)$$

resulting in $\Phi_i(q_i^{MLP_{nn}})$, being computed in two steps: 1) the cell node limiter coefficient, $\Psi_{i,n}^{nn}(q_{i,n})$, is computed at each node, *n*, that is a vertex of cell *i*, (equation 21) and 2) the cell-limiter coefficient is computed as the minimum of those cell node limiter coefficients (equation 20). In equation 21 the quantities, $q_{i,n}^{max(ncn)}$ and $q_{i,n}^{min(ncn)}$ are the maximum and minimum values of the node cell-neighbor stencil, *ncn*, illustrated in figure 9, and $q_{i,n}$ is the reconstruction of *q* to

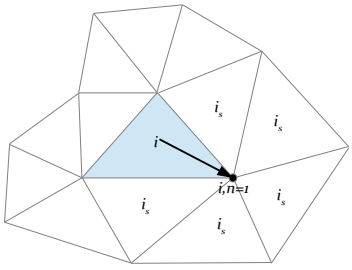

***Figure 9:** The node cell-neighbor stencil of cells sharing the n=1 node used to compute* $q_{i,1}^{max(ncn)}, q_{i,1}^{min(ncn)}$.

each node, *n*, of cell, *i*, based on an unlimited form of Fromm's scheme, i.e.,

$$q_{i,n} = q_i + \overline{\nabla q_i} \cdot \vec{r}_{in} \quad (22)$$

where $\phi_{i,n}$, is computed at each node, *n*, of cell *i*, by using equation 19, instead of at each face midpoint, *f*, of cell *i*. However, since $\Phi_i(q_i^{MLP_{nn}})$ was derived to use the *nn* stencil, which is not currently implemented in the code, it has been modified for use with the *fn2* stencil, $\Phi_i(q_i^{MLP_{fn2}})$, as illustrated in figure 10,

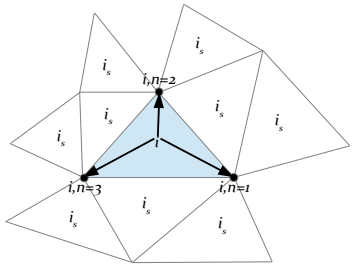

***Figure 10:** The fn2 stencil cells and nodes that participate in the construction of the* $\Phi_i(q_i^{MLP_{fn2}})$, *cell-limiter coefficients for the cell i where* $N_{i,nodes} = 3$.

where the terms $q_{i,n}^{max(ncn)}$ and $q_{i,n}^{min(ncn)}$ of equation 21 have been replaced with $q_i^{max(fn2)}$ and $q_i^{min(fn2)}$ respectively, to yield

$$\Phi_i(q_i^{MLP_{fn2}}) = min(1, [\Psi_{i,n}^{fn2}(q_{i,n}), n=1 \rightarrow N_i, nodes]) \quad (23)$$

$$[\Psi_{i,n}^{fn2}(q_{i,n}) = min\left(1,\ \begin{cases} \phi_{i,n}\left(\dfrac{q_i^{max(fn2)} - q_i}{q_{i,n} - q_i}\right), if\ (q_{i,n} - q_i) > 0 \\ \phi_{i,n}\left(\dfrac{q_i^{min(fn2)} - q_i}{q_{i,n} - q_i}\right), if\ (q_{i,n} - q_i) < 0 \\ 1 \qquad\qquad\qquad if\ (q_{i,n} - q_i) = 0 \end{cases}\right)], n = 1 \rightarrow N_{i,nodes}. \quad (24)$$

The resulting gradient limiter approach of equations 19, 23, and 24, while not a strict implementation of the $\Phi_i(q_i^{MLP_{nn}})$ limiter, has been found to be much more robust than the face-based limiter originally supplied by the NRA and less dissipative then the heuristic pressure limiter augmented stencil-based limiter approach of equations 19, 20, and 21.

Viscous Flux Cell Face Gradient Construction:

The computation of the viscous flux requires that the cell-face average of the primitive variables be computed, where the primitive variables in this case are

$$q = \left(\frac{\rho_1}{\rho}, \ldots, \frac{\rho_{ncs}}{\rho}, \rho, u, v, w, T, k, \omega\right) \quad \text{for thermal equilibrium, or}$$

$$q = \left(\frac{\rho_1}{\rho}, \ldots, \frac{\rho_{ncs}}{\rho}, \rho, u, v, w, T_{ve}, T, k, \omega\right) \quad \text{for thermal nonequilibrium}$$

as well as the cell-face average gradient $\overline{\nabla q}_f$. Hasselbacher [31] observed that computing $\overline{\nabla q}_f$ as a simple average of the face neighbor cell-average gradients, i.e.,

$$\overline{\nabla q}_f = \frac{\left(\overline{\nabla q}_i + \overline{\nabla q}_j\right)}{2} \quad (25)$$

leads to odd-even decoupling causing him to introduce face-derivative augmentation. Hasselbacher suggested two methods to accomplish this augmentation: the so-called, edge-normal (EN) and face-tangent (FT) cell-face gradient methods. More recently, Nishikawa [16] has suggested a new approach where the Hasselbacher's augmentation terms are shown to be analogous to damping terms. Nishikawa discretized a hyperbolic model for diffusion with an advection scheme, and derived a novel scheme for diffusion from the result. The resulting hyperbolic diffusion scheme, has a form that is very similar to the face-tangent augmented cell-face gradient method, and has a guiding principle based on the observation that the discretized scheme, being hyperbolic, can be treated using the formalisms developed to upwind the inviscid flux terms. The use of these upwinding formalisms, in turn, leads to a method in which damping terms arise naturally thereby resulting in a scheme that is consistent, damps high-frequency errors, and can be made 2nd-order accurate when applied to a cell-centered, finite-volume scheme. The edge-normal augmented cell-face gradient method is the method originally implemented in the NRA-supplied code. However, when the edge-normal and face-tangent augmented cell-face gradient methods were studied in [11,15,16], the face-tangent method was found to be preferable to the edge-normal method. Moreover, in [11], the observation was made that, in many cases, a converged solution could only be obtained when the face-tangent augmented face-gradient method was used. However, insight can be gained by comparing the edge-normal, face-tangent and hyperbolic-reconstruction cell-face gradient methods. Therefore, for the sake of completeness, the edge-normal method is also described. The edge-normal augmented cell-face gradient method, as defined by Hasselbacher, is

$$\widehat{\nabla q}_f^{EN} = \overline{\nabla q}_f - \left[\overline{\nabla q}_f \cdot \hat{e}_{ij} - \frac{\left(q_j - q_i\right)}{|\vec{e}_{ij}|}\right] \hat{e}_{ij} \quad (26)$$

where, referring to figure 2, $\vec{e}_{ij}$ is a vector drawn from cell-center *i* to cell-center *j* and $\hat{e}_{ij}$ is its unit vector. The face-tangent augmented cell-face gradient method, as defined by Hasselbacher, is

$$\widehat{\nabla q}_f^{FT} = \overline{\nabla q}_f - \left[\overline{\nabla q}_f \cdot \hat{e}_{ij} - \frac{\left(q_j - q_i\right)}{|\vec{e}_{ij}|}\right]\left(\frac{\hat{n}_f}{\hat{n}_f \cdot \hat{e}_{ij}}\right). \quad (27)$$

As mentioned above, Nishikawa considered the augmentation terms (the bracketed terms in equations 26 and 27) to be damping terms, and further observed that the face-tangent method damping term leads to a more robust scheme on highly skewed meshes due to the dependence on $1/(\hat{n}_f \cdot \hat{e}_{ij})$. The increased robustness results from the fact that as skewness increases, $\hat{n}_f \cdot \hat{e}_{ij}$ decreases, thereby increasing damping. When Nishikawa's hyperbolic diffusion based approach is applied to a cell-centered, finite-volume scheme, it results in a hyperbolic, reconstruction (HR) based cell-

face-gradient method that includes a damping term that arises naturally due to an upwind method being used to discretize the construction of the cell face gradient. This hyperbolic-reconstruction based cell-face gradient method has the form

$$\widehat{\nabla q_f}^{HR} = \overline{\nabla q_f} \; + \; \alpha \left(\frac{\hat{n}_f}{|\vec{e}_{ij} \cdot \hat{n}_f|} \right) \left(q_f{}^R - q_f{}^L \right) \quad (28)$$

where α is a damping coefficient and $q_f{}^L$ and $q_f{}^R$ are the left and right higher-order-reconstructed viscous face state variables. These state variables are reconstructed using Fromm's scheme where

$$q_f{}^L = q_i + \overline{\nabla q_i} \cdot \vec{r}_{if} \quad (29)$$

$$q_f{}^R = q_j + \overline{\nabla q_i} \cdot \vec{r}_{jf}. \quad (30)$$

Recently Jalai et al. [32] analyzed the stability and accuracy of nine cell-centered, finite-volume, cell-face gradient methods and reported that Nishikawa's scheme, with $\alpha = 4/3$, was the preferred scheme for computing the cell-face gradient, based on stability and accuracy analyses.

Implicit Scheme Enhancements:

The implicit schemes implemented in the NRA-supplied code were a point-implicit LU-SGS scheme and a point-implicit, matrix-free version of the symmetric Gauss-Seidel scheme (SGS) [18]. Testing subsequent to receiving the code revealed that while the scheme converged reasonably well on some inviscid flow problems, convergence and robustness were poor for hypersonic turbulent flow problems. Therefore, three changes were made to the implicit solver to improve its convergence rate and robustness. First, the linearization of the inviscid and viscous fluxes were improved. The LU-SGS scheme and the matrix-free SGS schemes both linearize the inviscid and viscous fluxes by approximating their Jacobians with their respective spectral radii. Second, implicit boundary conditions were implemented using an operator-overloaded implicit boundary condition methodology, and third, partition coupling was introduced by introducing inter-partition communication of the provisional updates between the forward sweeps of the *L* and backward sweep of the *U* resulting in a data-parallel version of the matrix-free and matrix-based SGS schemes. The derivation of the *LU* decomposition begins with writing the delta form of the backward Euler implicit scheme for solving the governing equations as

$$A_i^n \Delta Q_i^n = R_i \quad (31)$$

where R_i is the discrete steady state residual, A_i is the Jacobian and ΔQ_i^n the update are defined as

$$\Delta Q_i^n = Q_i^{n+1} - Q_i^n \quad (32)$$

and

$$A = \frac{V_i}{\Delta t_i} I - \left(\frac{\partial R_i}{\partial Q} \right) \quad (33)$$

and the steady state residual for a face-based data structured of a cell-centered scheme is the sum over the faces (ij) that define cell *i*

$$R_i = \sum_j F_{ij}. \quad (34)$$

A can be decomposed into the sum of lower (*L*), upper (*U*) and Diagonal (*D*) matrices

$$A = L + U + D \quad (35)$$

where

$$L = -\sum_j \left(\frac{\partial F_{ij}}{\partial Q_i} \right), \quad D = \left[\frac{V_i}{\Delta t_i} I + \sum_j \left(\frac{\partial F_{ij}}{\partial Q_i} \right) \right], \quad U = \sum_j \left(\frac{\partial F_{ij}}{\partial Q_j} \right). \quad (36)$$

The derivation of the matrix-based and matrix-free versions of the SGS scheme are discussed in detail in [18] but we summarize here that the k^{th} subiteration of the matrix-based form of the SGS scheme is written as a forward sweep of

$$\Delta Q_i^{k+1/2} = \left(D + L \right)^{-1} \left[R_i - U \Delta Q_i^k \right] \quad (37)$$

followed by a backward sweep of

$$\Delta Q_i^{k+1} = (D+U)^{-1}[R_i - L\,\Delta Q_j^{k+1/2}]. \quad (38)$$

Whereas the k^{th} subiteration of the matrix-free form of the SGS scheme is written as

$$\Delta Q_i^{k+1/2} = D_i^{-1}\left[R_i - \frac{1}{2}\sum_{j:\,j\in L_i}\left(\Delta F_j^{k+1/2} - \lambda_{ij}\Delta Q_j^{k+1/2}\right)s_{ij} - \frac{1}{2}\sum_{j:\,j\in U_i}\left(\Delta F_j^{k} - \lambda_{ij}\Delta Q_j^{k}\right)s_{ij}\right] \quad (39)$$

followed by a backward sweep of

$$\Delta Q_i^{k+1} = D_i^{-1}\left[R_i - \frac{1}{2}\sum_{j:\,j\in L_i}\left(\Delta F_j^{k+1} - \lambda_{ij}\Delta Q_j^{k+1}\right)s_{ij} - \frac{1}{2}\sum_{j:\,j\in U_i}\left(\Delta F_j^{k+1/2} - \lambda_{ij}\Delta Q_j^{k+1/2}\right)s_{ij}\right] \quad (40)$$

where λ_{ij}, is the sum of the spectral radii of the inviscid and viscous fluxes at the cell face between cells *i* and *j*, and ΔF_j^k, $\Delta F_j^{k+1/2}$, and ΔF_j^{k+1} are generally referred to as "flux increment" terms. These flux increment terms are defined in [18] as

$$\Delta F_j^{k+1/2} = F\left(Q^k + \Delta Q^{k+1/2}\right) - F\left(Q^k\right) \quad (41)$$

$$\Delta F_j^{k+1} = F\left(Q^{k+1/2} + \Delta Q^{k+1}\right) - F\left(Q^{k+1/2}\right) \quad (42)$$

Therefore, the k^{th} subiteration of the matrix-free SGS scheme only requires the spectral radii of the inviscid and viscous fluxes and the diagonal of the flux Jacobian, thereby allowing the scheme to be matrix-free and making it inexpensive from a storage and, in its calorically perfect gas form, operation count perspective. However, the flux increment terms must be computed after each forward and backward sweep, which adversely affects the cost of the matrix-free approach when computing calorically imperfect flows. When solving a flow where the gas is modeled as calorically perfect, the cost of computing the flux increment terms is not onerous. However, when solving a flow where the gas is modeled as calorically imperfect, the thermodynamics of the provisionally updated solution after the forward and backward sweep must be computed prior to computing the flux increment terms. The cost of this thermodynamics variable update, which occurs twice per iteration, is large due to the Newton solve used to update the static temperature. Moreover, the spectral radii approximation of the flux Jacobians was found to be the root cause of the poor convergence and robustness of the matrix-free scheme. Therefore, the spectral-radius-based 1st-order inviscid flux Jacobian was replaced with an exact linearization of a 1st-order inviscid flux derived using automatic differentiation (via modern FORTRAN's operator overloading capability) and the spectral-radius-based viscous flux Jacobian was replaced with a hand linearization of a thin-layer Navier-Stokes approximation of the viscous flux to form a matrix-based SGS scheme. The end result of these modifications was a significant net win for the matrix-based SGS scheme. This occurred for two reasons: 1) a cost reduction of the matrix-based SGS scheme realized, by taking advantage of the requirement that the flux Jacobians must be stored, by updating the flux Jacobians periodically based on an update frequency algorithm rather than every time step/cycle and 2) the elimination of the need to compute the flux increments twice for each subiteration of the SGS scheme for every time step/cycle. The Jacobian update frequency algorithm decreases the frequency of the evaluation of the flux Jacobian as a function of convergence of the steady-state residual. In addition, implicit boundary conditions, also based on operator-overloading automatic differentiation, were implemented and semi-implicit partition coupling was introduced through inter-partition communication of the provisional updates after the forward and backward sweeps of the SGS scheme.

Physical Modeling Extensions:

The unstructured-grid solver delivered at the conclusion of the NRA contained a subset of the thermodynamic, transport, chemical kinetic and turbulence modeling capabilities available in the structured-grid solver. However, because the unstructured-grid solver was written such that it shares all of the source term routines with the structured-grid solver, only minor modifications were required to make the additional chemical kinetic and turbulence models operational. However, because the thermal nonequilibrium capability did not exist in the structured-grid solver when the code was provided to to the NRA contractor, more extensive modifications were required to add that capability into the unstructured-grid solver.

y^+ Adaptive Turbulent Wall Boundary Condition:

The current state-of-the-art of commercial unstructured grid generation tools provides the user limited control of the spatial variation of the wall normal distance from the 1st-interior grid point to the wall. This can be problematic when dealing with viscous flows around geometries having large scale disparities. Hypersonic vehicle geometries typically have small-scale features, such as the diameter of a blunt leading edge, that require a wall-normal grid spacing that may be orders of magnitude smaller than at other locations in the grid. Therefore, the grid generation tool user is faced with a conundrum where they need to specify a wall spacing that is sufficient to adequately resolve wall shear and

heat transfer near the small scale features without drastically over resolving the wall flow in other locations. Turbulent flows exacerbate this problem because the turbulence model solve-to-the-wall boundary conditions require a wall-normal grid spacing that result in a y^+_{wall} <= 2.5 from a solver robustness point of view and <= 1.0 from an accuracy point of view. This maximum allowable y^+_{wall} requirement can be alleviated by employing a wall-matching-function boundary condition [33], which extends the allowable y^+_{wall} into the log-law range. However, if a grid has a y^+_{wall} that varies from viscous sublayer to log-law values, the wall-matching function will become inaccurate where the y^+_{wall} is within the viscous sublayer and the solve-to-the-wall boundary condition will become inaccurate, and possibly unstable, where the y^+_{wall} is bigger than 2.5. However, the usable y^+_{wall} range for the wall-matching-function boundary condition was extended in the VULCAN-CFD code by blending viscous sublayer behavior velocity behavior, $u^+=y^+$, with the log-law such that the resulting wall-matching function does a reasonable job for y^+_{wall} > 2.5. However, for y^+_{wall} < 2.5, it is still preferable to use a strict solve-to-the-wall boundary condition. Therefore, a y^+ adaptive turbulent wall boundary condition methodology was developed and implemented in the structured-grid and unstructured-grid solvers, which switches between solve-to-the-wall and wall-matching-function boundary conditions depending on the y^+_{wall} of each wall cell. This capability is realized using a 2 step process. The first step enforces the solve-to-the-wall boundary condition for a given wall cell computing the y^+_{wall} based on the current wall tangent velocity, wall molecular viscosity and wall density. If the resultant y^+_{wall} is < 2.5, then the wall cell is tagged as a solve-to-the-wall cell, if y^+_{wall} >= 2.5, then that wall cell is tagged as a wall-matching function cell. The second step then solves the wall-matching function for all the cells that were tagged as wall-matching-function cells also computing the wall face shear stress, the wall heat flux, as well as the wall adjacent cell center turbulent kinetic energy and the turbulent specific dissipation rate.

Results and Discussion

The unit test cases to follow involve the computation of turbulent high speed flow over a flat plate. These flat plate cases were computed using the unstructured-grid solver for Mach 6, Mach 2 and Mach 2.91 conditions. The Mach 6 condition will be used to examine the convergence behavior of the implicit scheme and to make some general recommendations as to how the implicit scheme should best be configured for solving turbulent high speed flow. The Mach 2 case is intended to demonstrate and examine the solver's ability to compute turbulent boundary layer flows on mixed cell type prismatic unstructured grids when using a solve-to-the-wall boundary condition by comparing the computed results with the turbulent law-of-the-wall. The Mach 2.91 flow turbulent flat plate case is intended to demonstrate and compare turbulent boundary layer flow computed using the y^+ adaptive turbulent wall boundary condition with the turbulent law-of-the-wall. The simulation of the flow through a realistic scramjet related geometry at representative conditions will then be performed for the 75% scale HIFiRE 7 Rectangular to Elliptic Shape Transition (REST) scramjet engine flow path that was free jet tested in the University of Queensland (UQ) T4 shock tunnel [19]. This computation will be performed for the zero degree angle of attack tare case only. The computed static pressure distributions on the body and cowl walls will be compared with the experimental data.

Testing of the inviscid and viscous flux enhancements as well as the implicit scheme improvements was performed for thermally perfect and calorically perfect turbulent flows over a flat plate using a mixed cell type grid. In addition, testing of the y^+ adaptive turbulent wall boundary condition was performed for Mach 2.91 calorically perfect turbulent flow over a flat plate using a hexahedral grid. Figure 11 presents the unstructured grid generated for a 0.455–meter flat plate that was used to test the convergence behavior of the implicit scheme and to validate the unstructured-grid solver against the turbulent boundary layer law-of-the-wall. This grid was a mixed cell type grid that had 227,500 cells consisting of a mixture of 59,112 tetrahedra, 1283 pyramids, 167,079 prisms and 0 hexahedra.

Implicit Scheme Enhancements:

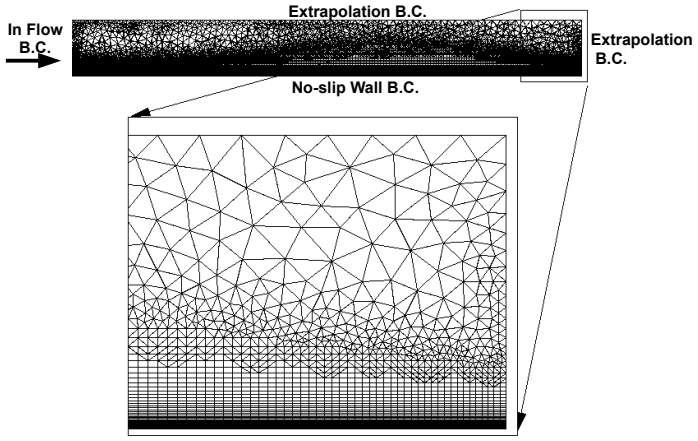

Figure 11: Unstructured grid used for the Mach 6 and Mach 2 turbulent flat plate test cases, and the axial station of the boundary layer profile extraction used for comparison with the turbulent law-of-the-wall.

The enhancements to the implicit scheme were tested by computing thermally prefect, chemically frozen, turbulent flow of air over a flat plate with freestream conditions of, Mach 6, static pressure, $P_{ref}=2100.0$ Pascals , static temperature, $T_{ref}=63.01$ Kelvin , and unit Reynolds number, $Re_{ref}=2.64 \times 10^{7}/\mathrm{m}$ with the wall treated as an isothermal (335.83 Kelvin), no-slip, solve-to-the-wall boundary condition. The governing equations were solved in a fully coupled manner, with local time stepping and the CFL number being linearly varied from 0.1 to 250 over time steps 1 to 500. The Wilcox (1998) $k-\omega$ two-equation turbulence model [34] was used to compute the Reynolds stresses and Reynolds heat flux (Pr_t=0.9). The cell-average gradients were computed using weighted linear least squares with the *fn2* stencil. The inviscid fluxes were computed using the LDFSS scheme with the higher-order cell-face states being computed using UMUSCL, $\kappa=1/3$, with the cell-average gradients limited using the $\Phi_i(q_i^{MLP_{fn2}})$ gradient limiter and the van Leer function. The viscous fluxes were computed using the Nishikawa cell-face gradient method. Convergence was achieved by "freezing" the gradient limiter after 200 time steps or once the residual L_2 norm had dropped 4 orders of magnitude to prevent convergence stalling due to limiter "ringing". The computations were stopped when the residual L_2 norm had dropped 12 orders of magnitude. Three studies were performed. The first study used a 6 partition decomposition of the grid to compare the matrix-free SGS scheme without implicit boundary conditions with the matrix-based SGS scheme with implicit boundary conditions. The second study also used the same number of partitions to examine the dependence of the convergence rate on the number of linear-solve subiterations for the matrix-free SGS without implicit boundary conditions and the matrix-based SGS scheme with implicit boundary conditions. The third study used 1, 2, 4, 8, 16 and 24 partition decompositions of the grid to compare the effectiveness of the matrix-based SGS scheme with inter-partition coupling in reducing the sensitivity of the convergence behavior to the number of partitions.

Figure 12 presents a comparison of the convergence between the matrix-free SGS scheme, using explicit boundary conditions, labeled Matrix Free, and the matrix-based SGS scheme, using implicit boundary conditions, labeled Matrix Based. Four subiterations of the linear-solver were run for both schemes. This number of subiterations was chosen because that was the maximum number that the matrix-free SGS scheme handle without diverging. Figure 12 shows that the matrix-based SGS and implicit boundary condition combination resulted in a speed up factor of 2.6 in terms of time steps/cycles and 4.9 in terms of CPU time over the matrix-free SGS scheme. The iterative speed up was due to the improved convergence rate provided by the improved approximation to the flux Jacobian of the matrix-based SGS scheme and the CPU time improvement is due to the elimination of the thermodynamic and transport property evaluation required by the flux increment in the matrix-free SGS scheme, and by taking advantage of the Jacobian "freezing" that is possible with the matrix-based SGS scheme.

The effect of increasing the number of linear solve subiterations for the matrix-based SGS scheme is presented in figure 13. Remembering that the matrix free SGS scheme was unstable for when more than 4 subiterations were applied, figure 13 shows that the matrix-based SGS scheme has no such limitation and that increasing the number of subiterations improved the convergence rate for values up to 15.

The dependence of the convergence rate of the matrix based SGS scheme, with inter-partition coupling, on the number of partitions for a fixed number of linear solve subiterations (in this case 10) is shown in figure 14. The figure shows that convergence slowly degraded as the number of partitions increased. Figure 15 presents the 24 CPU/partition case run with 10, 15, 20 and 25 subiterations, demonstrating that increasing the number of subiterations improves the convergence rate during the latter portion of the solution process. Furthermore, it is important to note that when the 4, 8, 16 and 24 CPU/partition cases were run without linear solver inter-partition coupling, the matrix-based SGS scheme became unstable for 8 or more partitions. This demonstrates that inter-partition coupling stabilizes the matrix-based SGS scheme.

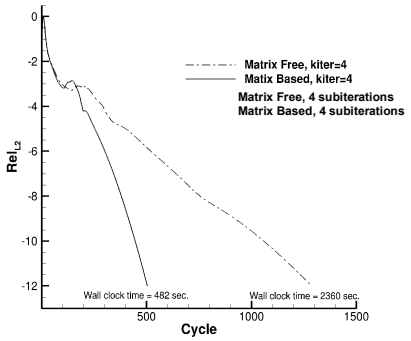

Figure 12: Convergence improvement due to the improved inviscid and viscous jacobian approximations and adding implicit boundary conditions.

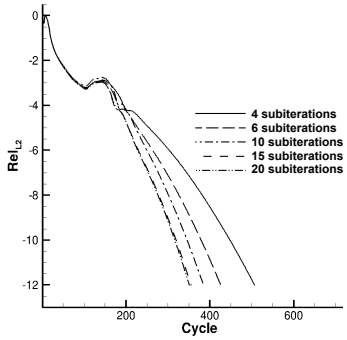

Figure13: The effect of the number of linear solve subiterations on convergence rate when using the matrix-based SGS scheme.

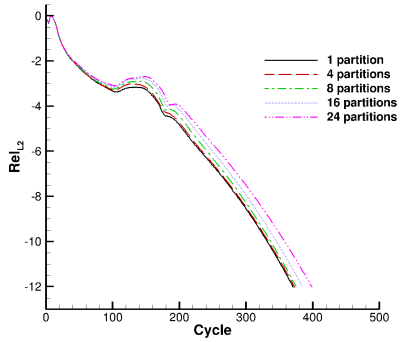

Figure 14: The effect of increasing the number of partitions, for 10 linear-solve subiterations, on convergence rate with inter-partition coupling using the matrix-based SGS scheme.

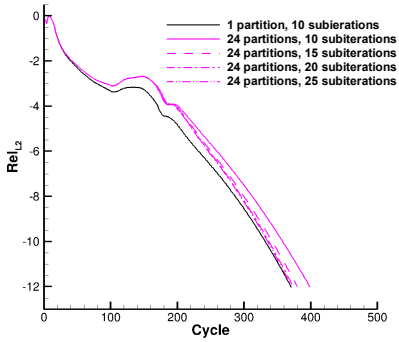

Figure 15: The effect of increasing the number of linear-solve subiterations, for 24 partitions, on the convergence behavior with inter-paritition coupling using the matrix-based SGS scheme.

Based on the results shown in figures 13-15, 10-15 coupled linear solve subiterations are recommended to obtain a robust matrix-based SGS scheme when parallel processing. However, because figure 15 also indicates that sensitivity to the number of partitions remains during the initial part of the solution process, the implementation of a multi-color Gauss-Seidel scheme similar to the implementation found in FUN3D [35] should be considered.

Comparison of a Mach 2 Turbulent Flat Plate Solution With the Turbulent Law-of-the-wall:

The unstructured mixed cell type grid shown in figure 11 was also used to simulate calorically perfect turbulent flow over a flat plate with freestream conditions of Mach 2, $P_{ref}=10{,}374.2$ Pascals, $T_{ref}=165.64$ Kelvin, $Re_{ref}=1.0\times10^{7}/\text{m}$ and $\gamma_{ref}=1.4$ with the wall treated as an adiabatic no-slip solve-to-the-wall boundary condition. The same inviscid and viscous flux construction methods, turbulence model, and CFL number schedule, as described for the Mach 6 turbulent flat plate case, were used for this case as well. The grid was decomposed into 6 partitions and with the number of subiterations set to 10. The objective of this case was to compare the boundary layer profile, far enough downstream to avoid leading edge effects, with turbulent law-of-the-wall. Figure 16 presents the convergence behavior and level of convergence obtained for this case. As in the Mach 6 case the $\Phi_i(q_i^{MLP_{fn2}})$ limiter was frozen once the residual L_2 norm had dropped 4 orders of magnitude and the computation was stopped when the residual L_2 norm had dropped 12 orders of magnitude. The boundary layer profile was extracted from the computational domain at the x=0.425–m axial station along with the wall shear stress at that location. Figure 17 shows profiles of the U-velocity component and the eddy viscosity ratio (the eddy viscosity normalized by the local molecular viscosity), indicating that the wall boundary layer has reached a turbulent state. The velocity profile was transformed with the van Driest II transformation [36] and the transformed velocity and wall-normal coordinate were then normalized with the wall friction velocity based on the wall shear stress at the x=0.425–m axial station. Figure 18 compares the transformed computed velocity profile with the turbulent law-of-the-wall. Agreement in the $u^+=y^+$ region is excellent with a slight deviation in slope with respect to the log-law region. This disagreement is dependent on the value of the von Karman's constant (κ) chosen when computing the log-law portion of the profile. The value of κ is generally accepted to be between $0.38<\kappa<0.42$ for smooth wall flows [37], with the $\kappa=0.4$ value used here being considered the "nominal" value.

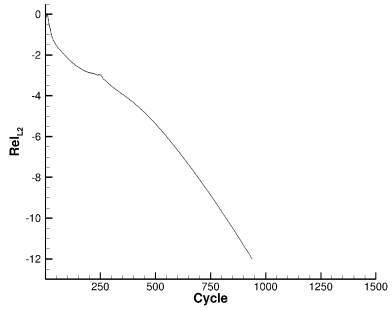

Figure 16: Convergence behavior of the matrix-based SGS scheme for the Mach 2 turbulent flat plate.

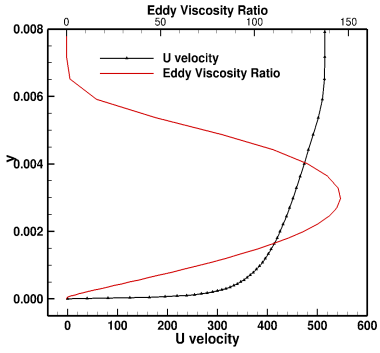

Figure 17: Computed U velocity and eddy viscosity profiles at the x=0.425–m axial station.

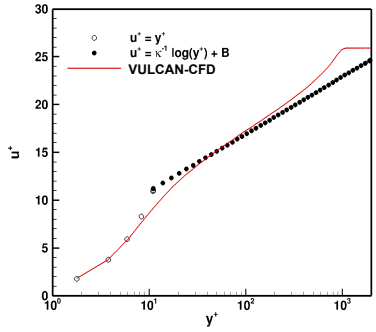

Figure 18: Comparison of computed velocity profile (line) with boundary layer turbulent law-of-the-wall theory (symbols) at the x=0.425–m axial station.

y^+ Adaptive Turbulent Wall Boundary Condition:

The y^+ adaptive turbulent wall boundary condition was tested using calorically perfect, turbulent flow over a flat plate with freestream conditions of, Mach 2.91, $P_{ref}=57{,}996.2$ Pascals, $T_{ref}=301.1$ Kelvin, $\gamma_{ref}=1.4$ and

$Re_{ref} = 3.67 \times 10^7 / m$ with the wall treated as an adiabatic no-slip y^+ adaptive turbulent wall boundary condition. The flat plate was 0.5588 meters long and was discretized using a grid containing 29,328 hexahedral cells with a wall normal first cell-center distance from the wall that varied along the length of the plate so as to produce increasing values of y^+_{wall} with increasing x. The governing equations were solved in a fully coupled manner using local time stepping with the CFL number being linearly varied from 0.1 to 200 over time steps 1 to 500. The Menter Baseline [39] two-equation turbulence model was used to compute the Reynolds stresses and Reynolds heat flux (Pr_t=0.9). All other options were kept the same as with the previous test cases. The grid was decomposed into 8 partitions for parallel processing with the number of subiterations set to 10. Convergence was achieved by "freezing" the gradient limiter once the residual L_2 norm had dropped 4 orders of magnitude and the computation was stopped when the residual L_2 norm had dropped 12 orders of magnitude. Figure 19 presents the computational hexahedral (hex) grid, the wall distribution of the axial shear stress and the y^+_{wall} using the distance from the wall cell face center to the cell-center of the wall cell. In addition, there are four axial stations shown where axial velocity profiles were extracted for comparison with the turbulent law-of-the-wall. Figures 20, 21, 22 and 23 present those profiles. The y^+ adaptive turbulent wall boundary condition was designed so that the crossover from a solve-to-the-wall boundary condition to a wall-matching-function boundary condition occurs when the wall y^+_{wall} >= 2.5. Examination of the y^+_{wall} distribution in figure 19 reveals that this occurs near x=0.1055–m. Scrutiny of the wall shear stress distribution in this vicinity reveals a small discontinuity that can be attributed to the abrupt change from the solve-to-the-wall boundary condition to the wall-matching-function boundary condition. Reduction of this discontinuity is a subject for future work.

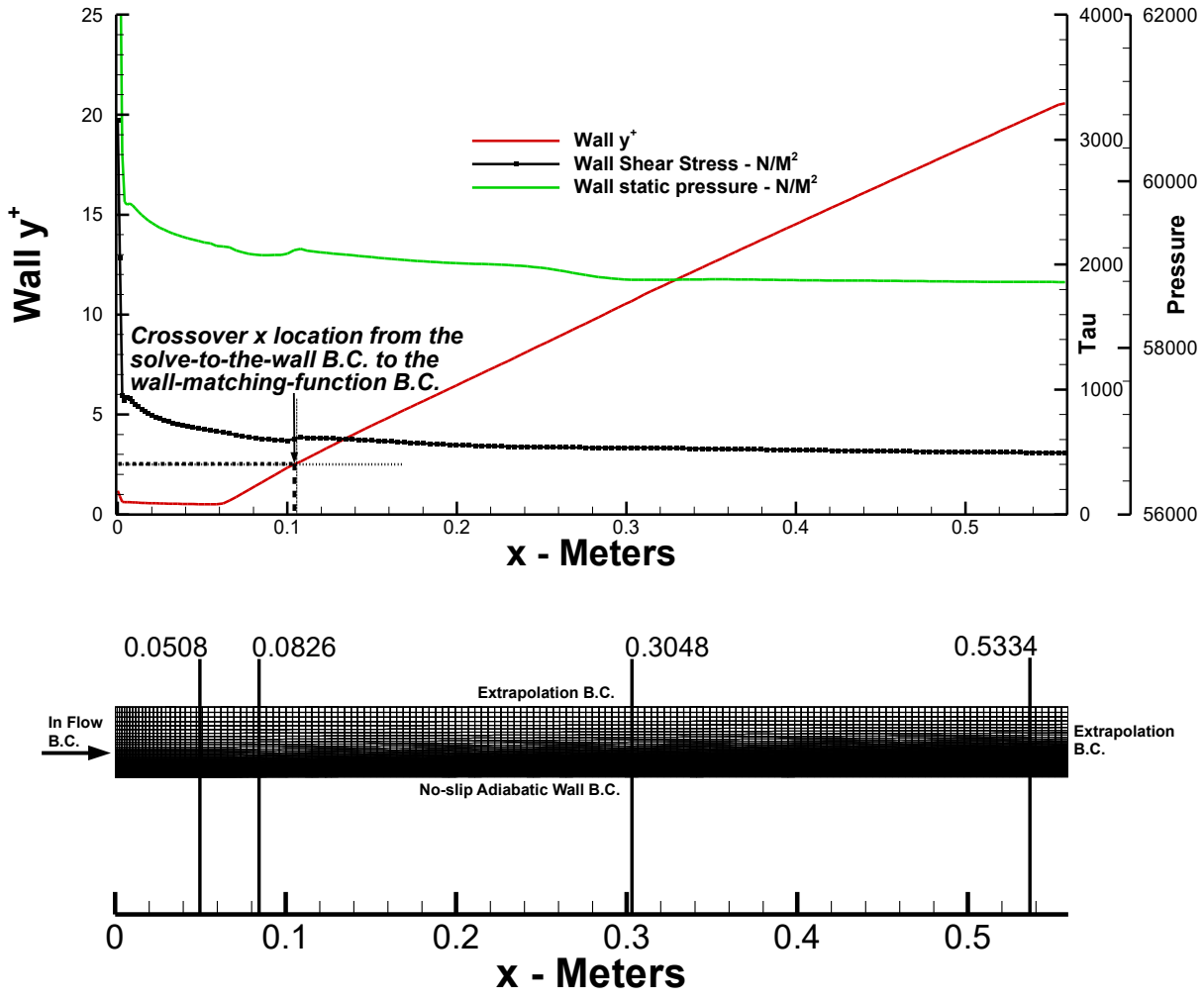

Figure 19: Unstructured Hex Grid and Wall solution for turbulent, calorically perfect, Mach 2.91 flow over a flat plate computed using the y+ adaptive turbulent wall boundary condition.

Examination of the profiles extracted upstream of the crossover location reveal that the solution extends down into the viscous sublayer with the near wall points closely matching the expected values given by the $u^+=y^+$ portion of the turbulent law-of-the-wall (open symbols). Agreement with respect to the log-law (filled symbols) portion of the profile is not as good, which is not too surprising, given that this close to the leading edge of the flat plate, the streamwise pressure gradient, as the wall pressure distribution presented in figure 19 demonstrates, is not zero. The reader is directed to [39] for a discussion of the effect of pressure gradient on the turbulent law-of-the-wall. However, examination of the profiles extracted downstream of the crossover at axial locations 0.3048–m and 0.5334–m, presented in figures 22 and 23 respectively, reveals that the profiles extracted where the wall-matching-function boundary condition were used are in much better agreement with the log-law. In addition, the $u^+=y^+$/log-law blending in the wall-matching-function boundary condition is visible in figures 22 and 23 because both profile y^+_{wall} values, 12.5 and 19.7 respectively, fall in the buffer layer. Moreover, profiles of the axial velocity and eddy viscosity ration are presented in figure 24 at axial location 0.5334–m to illustrate how well the boundary layer is resolved even when the y^+_{wall} is on the order of 20. Finally, the convergence history is presented in figure 25, showing that the convergence does not appear to have been adversely affected by the y^+ adaptive turbulent wall boundary condition.

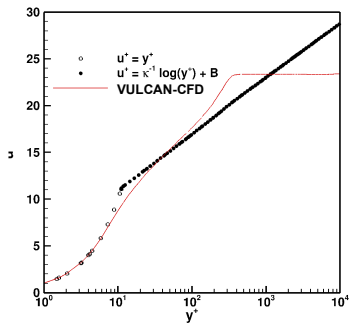

Figure 20: Comparison of a computed velocity profile (line) with the turbulent law-of-the-wall (symbols) at the x=0.0508–m axial station, where the solve-to-the-wall boundary condition was used.

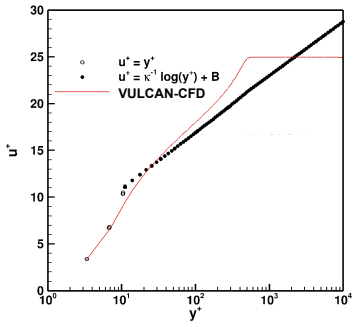

Figure 21: Comparison of a computed velocity profile (line) with the turbulent law-of-the-wall (symbols) at the x=0.0826–m axial station, where the solve-to-the-wall boundary condition was used.

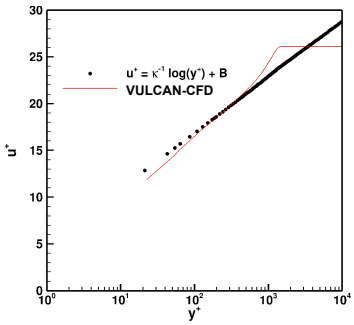

Figure 22: Comparison of a computed velocity profile (line) with the turbulent law-of-the-wall (symbols) at the x=0.3048–m axial station, where the wall-matching-function boundary condition was used.

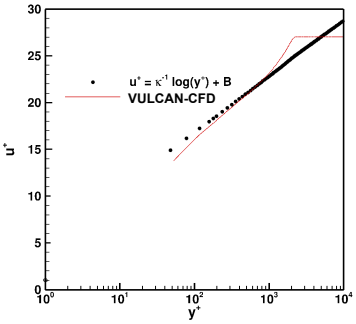

Figure 23: Comparison of a computed velocity profile (line) with the turbulent law-of-the-wall (symbols) at the x=0.5334–m axial station, where the wall-matching-function boundary condition was used.

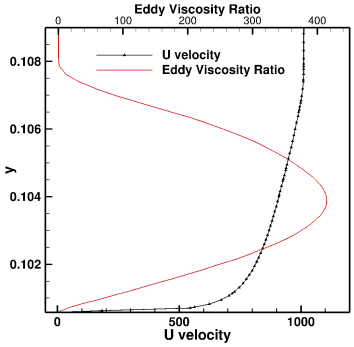

Figure 24: Computed U velocity and Eddy Viscosity Ratio profiles at the x=0.5334–m axial station, where the wall-matching-function boundary condition was used.

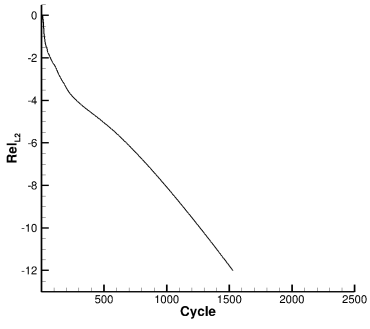

Figure 25: Convergence behavior of the matrix-based SGS scheme for the Mach 2.91, calorically perfect, turbulent flat plate when using the y⁺ adaptive turbulent wall boundary condition.

Simulation of the Shock Tunnel HIFiRE 7 REST Scramjet Engine Experiment:

The University of Queensland experimental test of a 75% scale replica of the HIFiRE 7 REST scramjet engine conducted in the T4 Stalker Tube [19] was simulated using the unstructured-grid solver. The zero degree angle-of-attack, tare (no fuel injection), test point was selected for simulation. The bilateral symmetry of the model geometry was exploited and a computational mesh for half of the REST scramjet flow path, shown in figure 25, was generated using Pointwise®. The surface grid was predominantly made up of triangles with quadrilaterals being used along the blunt leading edges and in the internal portion of the flow path. This surface grid was then marched normal to the wall surface into the interior of the computational domain to form a boundary layer grid made up of prisms and hexes. This boundary layer grid then transitioned into pyramids and tetrahedra resulting in a mixed cell type grid. The grid has a total of 44,568,851 cells consisting of 19,198,513 tetrahedra, 1,436,197 pyramids, 18,455,646 prisms, and 5,488,495 hexahedra, which were then decomposed into 768 partitions with ParMETIS.

The unstructured-grid solver was run with the same inflow/reference conditions used in [19] to perform their CFD simulation using the VULCAN-CFD structured-grid solver. These conditions were: $P_{ref} = 1675.0$ Pascals, $T_{ref} = 228.0$ Kelvin, and velocity, $U_{ref} = 2379\,\mathrm{m/s}$. As mentioned above, the zero degree angle-of-attack, tare case, was selected. However, it is important to note that the angel-of-attack convention reported in [19] is relative to the combustor centerline. The angle of attack relative to the x-axis, which runs parallel to the forebody plate, is 6 degrees, as shown in figure 26. This is the angle of attack that was run for the current study. A thermally perfect air gas mixture was used to simulate the test gas, which at the given reference conditions yields a Mach number of 7.845 and a unit Reynolds number of $Re_{ref} = 4.1 \times 10^6/\mathrm{m}$ The model surfaces were treated as no-slip, isothermal (300.0 Kelvin) walls, using the y^+ adaptive turbulent wall boundary condition. The Menter Baseline two-equation turbulence model was used to compute the Reynolds stresses and Reynolds heat flux (Pr_t=0.9). The inviscid fluxes were computed using the LDFSS scheme with the higher-order cell-face states being reconstructed using the UMUSCL, $\kappa = 1/3$, scheme with the cell-average gradients limited using the $\Phi_i(q_i^{MLP_{fn2}})$ gradient limiter and the van Leer function. The viscous fluxes were computed using the Nishikawa cell-face gradient method. The boundary layer trips were not modeled.

The governing equations were solved in a fully coupled manner with the matrix based SGS scheme with linear-solve inter-partition coupling and 10 subiterations. The computational domain was initialized to the reference conditions, and a 5-mm thick "initial boundary layer" was constructed by linearly blending the no-slip isothermal wall condition into the interior of the computational domain. The computation was run for 3000 cycles using a 1st-order advection scheme to establish the flow. The advection scheme was then switched to the 2nd-order scheme and the solution was run an additional 6000 cycles. The CFL number schedule is presented in figure 30. The gradient limiter was then frozen at cycle 9000, which can be seen in the residual plot in figure 30, as an abrupt drop in the residual. The solution was then run several thousand more cycles to make certain that it was stable with the frozen limiter. It was deemed "safe" to freeze the limiter at cycle 9000 because the mass flow error, surface integrated heat transfer, as well as the integrated forces and moments had all reached an asymptotic value. This approach resulted in the L_2 of the Residual converging approximately 5.5 orders of magnitude relative to its maximum value. Examination of the y^+_{wall} analysis reported by the y^+ adaptive boundary condition scheme as part of the solution process reveals that the y^+ adaptive turbulent wall boundary condition was an enabling technique in obtaining a solution on this computational grid. This analysis reported that there was a 3 order of magnitude variation of y^+_{wall} in the solution where the minimum, maximum, and area-weighted mean values of y^+_{wall} were 0.23, 247.0 and 4.17, respectively. The minimum value would have made the wall-matching-function boundary condition inaccurate and the maximum value would have made the solve-to-the-wall boundary condition unstable. The analysis also reported that 23.5% of the wall cells used the solve-to-the-wall boundary condition and 76.5% used the wall-matching-function boundary condition.

A Mach contour plot on the symmetry plane of the forebody and inlet is presented in figures 26 and 27. Figure 26 illustrates the small size of the forebody and cowl leading edges relative to the forebody boundary layer thickness. The figure also shows that the shocks are captured without significant oscillations. Figure 27, which superimposes the computational grid on the Mach contours, also shows that the forebody leading edge and forebody compression corner shocks are both captured with a small number of cells even where the grid is predominantly tetrahedral in nature. Finally, the the body side and inlet side wall static pressure distributions at the symmetry plane are presented in figures 28 and 29. The comparison is very reasonable and is slightly better than the solution reported in [19] that was obtained using the VULCAN-CFD structured-grid solver. A grid refinement study and comparison with the experimental Stanton number data is planned for the near future.

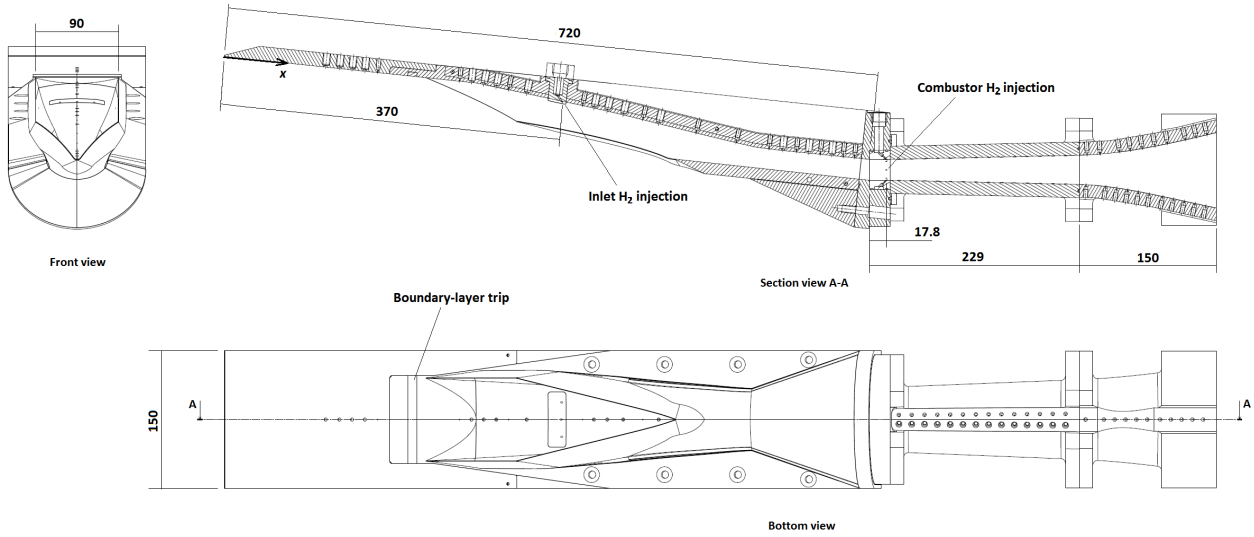

Figure 25: Schematic of HIFiRE 7 REST scramjet engine experimental model as installed in the University of Queensland T4 Stalker Tube, at 0 degrees angle of attack relative to the combustor centerline, 6 degrees angle-of-attack relative to the forebody plate surface (the labeled x-axis), reprinted with permission from reference 19.

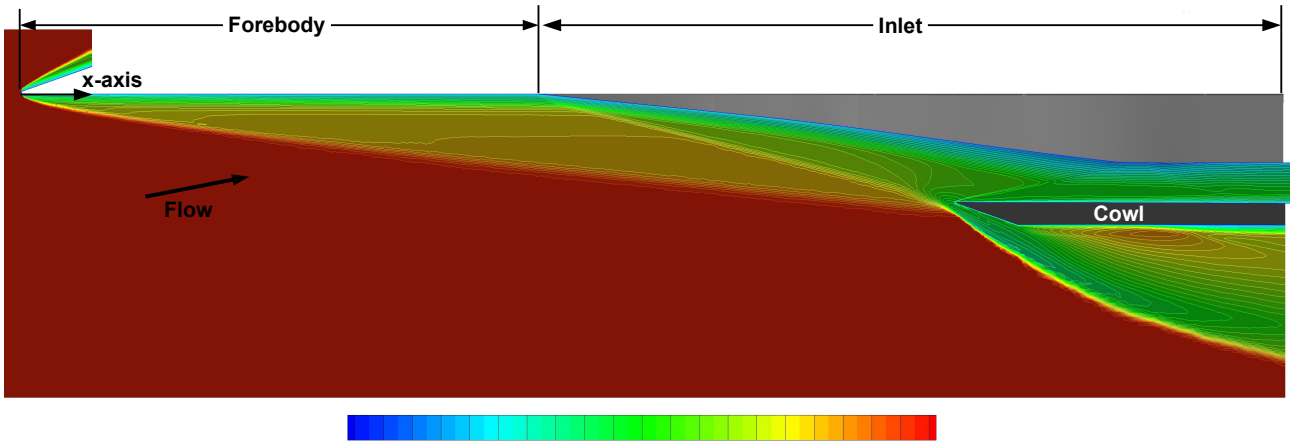

Figure 26: HIFiRE 7 REST forebody/inlet symmetry plane computational Mach contours, with flow at 6 degrees angle of attack relative to the forebody plate (x-axis) .

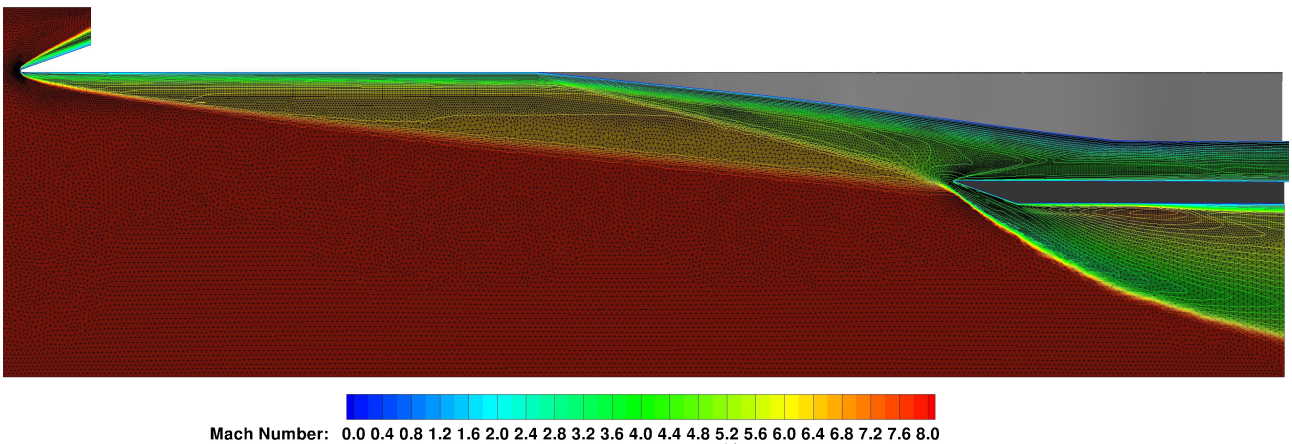

Figure 27: HIFiRE 7 REST forebody/inlet symmetry plane computational Mach contours with unstructured grid superimposed.

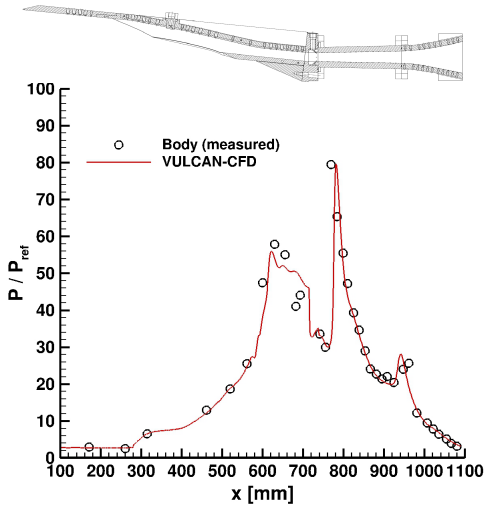

Figure 28: Comparison of the computed HIFiRE 7 flow path body side wall center-line pressure distribution with experimental data [19].

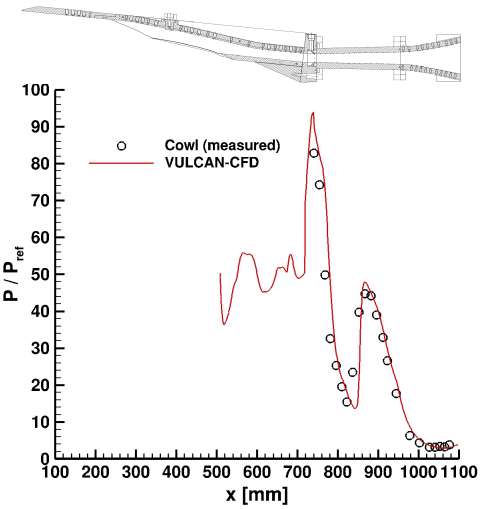

Figure 29: Comparison of the computed HIFiRE 7 flow path cowl side wall center-line pressure distribution with experimental data [19].

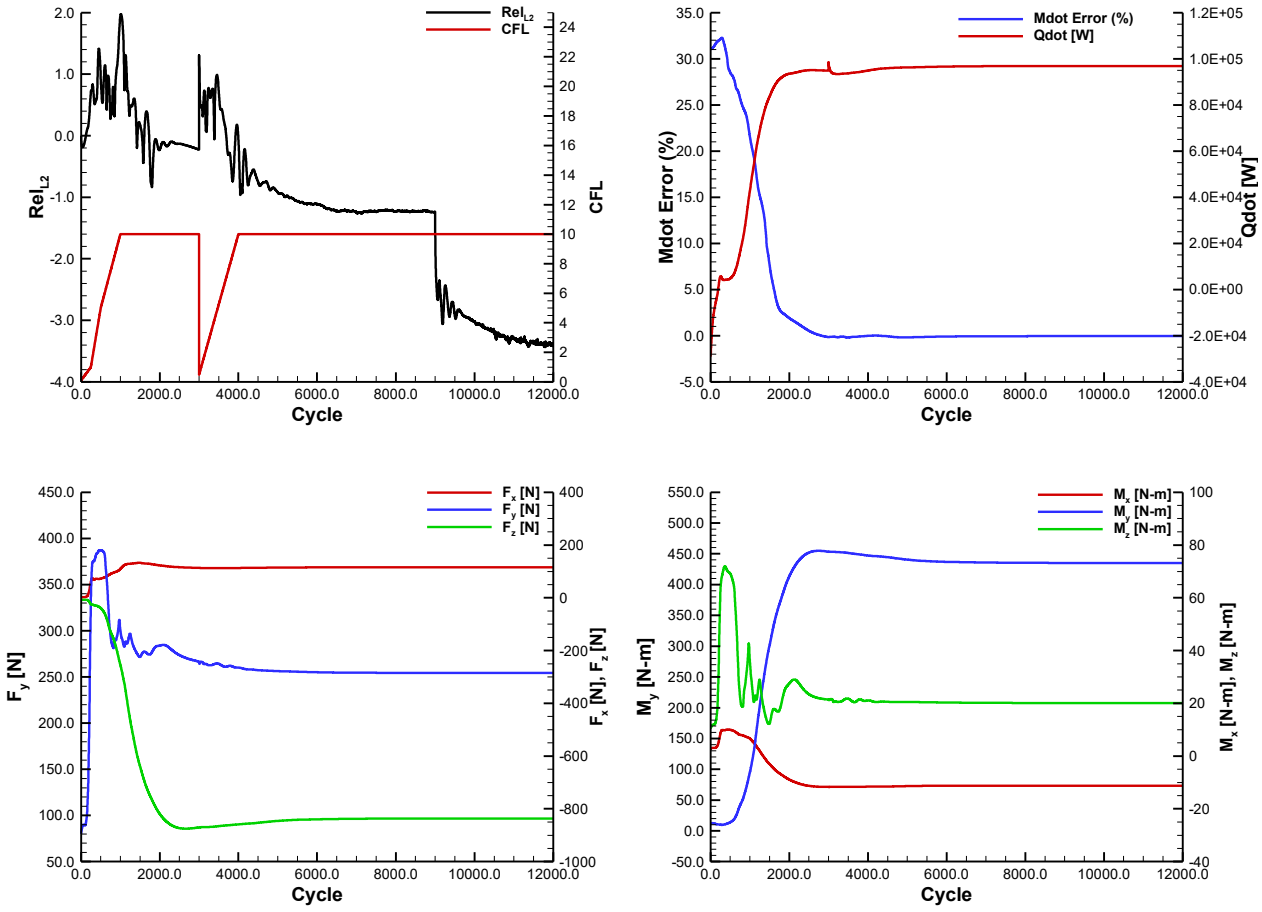

Figure 30: HIFiRE 7 flow path simulation convergence history of the L_2 of the residual, mass flow error, integrated wall heat transfer, forces and moments.

Conclusions

As mentioned in the introduction, the genesis of the computational kernel of the VULCAN-CFD unstructured-grid solver was a cell-centered, finite-volume, hybrid structured/unstructured-grid NRA developed code. However, when state-of-the-art non-hex dominant unstructured-grids were generated for realistic hypersonic vehicle geometries and simulated at "representative" hypersonic flow conditions, numerical accuracy, efficiency and robustness deficiencies as well as parallel scaling challenges were exposed. These accuracy, robustness and efficiency difficulties were alleviated by implementing the best-practices available in the current literature for 2nd-order accurate, cell-

centered, finite-volume, unstructured-grid schemes. In addition, great care has been taken to reuse the thermodynamic, transport, turbulence and chemical kinetic model subroutines and modules originally developed for the structured-grid solver such that all structured-grid solver modeling capabilities have been duplicated in the unstructured-grid solver. Moreover, new capabilities developed for the unstructured-grid solver, such as the y^+ turbulent wall boundary condition, have been also implemented in the structured-grid solver.

The successful implementation of the inviscid flux, viscous flux and implicit time stepping schemes as well as the development of the y^+ adaptive turbulent wall boundary condition scheme demonstrated in the flat plate test cases and their subsequent successful application to the HIFiRE 7 inlet geometry demonstrates that the code has reached a sufficient level of maturity that it can begin to be used by the greater community. Consequently, the unstructured-grid solver is being actively tested by the CFD application staff of the Hypersonic Airbreathing Propulsion Branch at the NASA Langley Research Center in support of projects of interest to NASA and the DOD. The promise of unstructured grids has already begun to be realized due to a substantial reduction in the time required to proceed from receipt of geometry to the completion of analysis. This has been primarily due to the dramatic reduction in the amount of time required to generate the computational grid afforded by the unstructured grid approach. The grid generation portion of the CFD process has decreased from on the order of a month, for pure structured grids, to less than two days, for unstructured grids. However, as mentioned in the introduction, the computational resource requirements embodied in the unstructured-grid paradigm have, as expected, proven to be significantly greater than those historically required by the structured-grid solver. Therefore, given the computational resource constraints that typically exist in restricted access computational environments, the time has come to incorporate the unstructured-grid spatially-elliptic flow solver capability into the VULCAN-CFD code's multi-region domain decomposition framework. This will allow the computational domain to be decomposed into a multi-region mix of multi-block structured-grid spatially-elliptic, multi-block structured-grid spatially-parabolic, and spatially-elliptic unstructured-grid regions that are solved sequentially thereby reducing the memory, processor and wall clock time requirements of the overall computation. Funding permitting, the development of this capability will be documented in a Part 2 sequel to this paper.

Acknowledgments

This work was supported by the Hypersonic Technology Project, through the Hypersonic Airbreathing Propulsion Branch of the NASA Langley Research Center.

References

1. Gnoffo, P.A., Wood, W.A., Kleb, B., Alter, S.J., Glass, C., Padilla J., Hammond, D. and White, J.A., "Functional Equivalence Acceptance Testing of FUN3D for Entry, Descent, and Landing Applications,"AIAA-2013-2558, 21st AIAA Computational Fluid Dynamics Conference, June, 2013.

2. Nompelis, I., Drayna, T. and Candler, G., "Development of a Hybrid Unstructured Implicit Solver for the Simulation of Reacting Flows Over Complex Geometries," AIAA-2004-2227, 34th AIAA Fluid Dynamics Conference and Exhibit, June, 2004.

3. Luke, E, "On Robust and Accurate Arbitrary Polytope CFD Solvers (Invited)," AIAA-2007-3956, 18th AIAA Computational Fluid Dynamics Conference, Miami, FL, June 2007.

4. Goldberg, U., "Hypersonic Turbulent Flow Predictions Using CFD++", AIAA-2005-3214, AIAA/CIRA 13th International Space Planes and Hypersonics Systems and Technologies Conference, May, 2005.

5. White, J.A. and Morrison, J.H., "A Psuedo-Temporal Multi-Grid Relaxation Scheme for Solving the Parabolized Navier-Stokes Equations," AIAA-1999-3360, 14th AIAA Computational Fluid Dynamics Conference, June, 1999.

6. Litton, D.K., Edwards J.R., and White, J.A., "Algorithmic Enhancements to the VULCAN Navier-Stokes Solver," AIAA-2003-3979, 16th AIAA Computational Fluid Dynamics Conference, June, 2003.

7. Spiegel, S.C., Stefanski, D.L., Luo, H., and Edwards, J.R., "A Cell-Centered Finite Volume Method for Chemically Reacting Flows on Hybrid Grids," AIAA-2010-1083, 48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition, Aerospace Sciences Meeting, July, 2010.

8. Spiegel, S.C., Stefanski, D.L., Luo, H., and Edwards, J.R., "A Regionally Structured/Unstructured Finite Volume Method for Chemically Reacting Flows," AIAA-2011-3048, 20th AIAA Computational Fluid Dynamics Conference, Fluid Dynamics and Co-located Conferences, July, 2011.

9. Diskin, B. and Thomas, J.L., "Comparison of Node-Centered and Cell-Centered Unstructured Finite Volume Discetizations: Inviscid Fluxes," AIAA-2010-1079, 48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition, January, 2010.

10. Sozer, E., Brehm, C. and Kiris, C.C., "Gradient Calculation Methods on Arbitrary Polyhedral Unstructured Meshes for Cell-Centered CFD Solvers," AIAA-2014-1440, 52nd AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition, January, 2014.

11. Schwöppe, A. and Diskin, B., "Accuracy of the Cell-Centered Grid Metric in the DLR TAU-Code," *New Results in Numerical and Experimental Fluid Mechanics VIII. Notes on Numerical Fluid Mechanics and Multidisciplinary Design*, Vol. 121, pp. 429-437, Springer, Berlin, Heidelberg, 2013.

12. Park, J.S. and Kim, C. "Mutli-dimensional Limiting Process for Finite Volume Methods on Unstructured Grids," *Computers & Fluids*, Vol. 65, Elsevier Ltd, 2012, pp. 8–24.

13. Burg, C.O.E., "Higher-Order Variable Extrapolation for Unstructured Finite Volume RANS Flow Solvers," AIAA-2005-4999, 17th AIAA Computational Fluid Dynamics Conference, June, 2005.

14. Diskin, B., Thomas, J.L., Nielsen, E., Nishikawa, H. and White, J.A., "Comparison of Node-Centered and Cell-Centered Unstructured Finite Volume Discetizations: Viscous Fluxes," AIAA-2009-0597, 39th AIAA Fluid Dynamics Conference, July, 2009.

15. Diskin, B., Thomas, J.L., Nielsen, E., Nishikawa, H. and White, J.A., "Comparison of Node-Centered and Cell-Centered Unstructured Finite Volume Discetizations: Viscous Fluxes," *AIAA Journal*, Vol. 48, No. 7, 2010, pp. 1326–1339.

16. Nishikawa, H., "Beyond Interface Gradient: A General Principle for Constructing Diffusion Schemes," AIAA-2010-5093, 40th AIAA Fluid Dynamics Conference, June, 2010.

17. Jameson, A. and S. Yoon, S., "Lower-upper Implicit Schemes With Multiple Grids for the Euler Equations," *AIAA Journal*, Vol. 25, No. 7, 1987, pp. 929–935.

18. Sharov, D. Luo. H., Baum, J.D., and Lohner, R., "Implementation of Unstructured Grid GMRES+LU-SGS Methods on Shared-memory, Cached-based Parallel Computers," AIAA-2000-0927, 38th AIAA Aerospace Sciences Meeting, January, 2000.

19. Chan, Y.K., Razzaqi, S. Smart, M.K. and Wise, D., "Freejet Testing of the 75%-scale HIFiRE 7 REST Scramjet Engine," AIAA-2014-2931, 19th AIAA International Space Planes and Hypersonic Systems and Technologies Conference, June, 2014.

20. Cabell, K., Hass, N., Storch, A., and Gruber, M., "HIFiRE Direct-Connect Rig (HDCR) Phase I Scramjet Test Results from the NASA Langley Arc-Heated Scramjet Test Facility," AIAA-2011-2248, 17th AIAA International Space Planes and Hypersonic Systems and Technologies Conference, April, 2011.

21. Edwards, J.R., "A low-diffusion flux-splitting scheme for Navier-Stokes calculations," *Computers & Fluids* Vol. 26, No. 6, 1997, pp. 635–659.

22. Toro, E.F., Spruce, M., and Speares, W., "Restoration of the contact surface in the HLL-Riemann solver," *Shock Waves,* Vol. 4, 1994, pp. 25–34.

23. Fromm, J.E. , "A method for reducing dispersion in convective difference schemes," *Journal of Computational Physics*, Vol. 3, 1968, pp. 176–189.

24. van Leer, B., "Towards the ultimate conservative difference scheme. V. A second-order sequel to Godunov's method," *Journal of Computational Physics,* Vol. 32, 1979, pp. 101–136.

25. Barth, T.J. and Jespersen, D. "the Design and Application of Upwind Schemes on Unstructured Meshes," AIAA-1989-0366, 27th AIAA Aerospace Sciences Meeting, January, 1989.

26. Venkatakrishnan, V., "On the Accuracy of Limiters and Convergence to Steady State Solutions," AIAA-1993-0880, 31st Aerospace Sciences Meeting, January, 1993.

27. van Albada, G.D., van Leer, B., and Roberts, W.W., "A comparative study of computational methods in cosmic gas dynamics," *Astronomy and Astrophysics*, Vol. 108, 1982, pp. 76–84.

28. Sweby, P.K. “High resolution schemes using flux-limiters for hyperbolic conservation laws,” *SIAM Journal of Numerical Analysis*, Vol. 21, 1984, pp. 995–1011.

29. Koren, B., "A robust upwind discretisation method for advection, diffusion and source terms", *Numerical Methods for Advection–Diffusion Problems*, Braunschweig: Vieweg, 1993, p. 117.

30. Gnoffo, P.A., “Updates to Multi-Dimensional Flux Reconstruction for Hypersonic Simulations on Tetrahedral Grids,” AIAA-2010-1271, 48th AIAA Aerospace Sciences Meeting, January, 2010.

31. Hasselbacher, A. and Blazek, J., “On the Accurate and Efficient Dscretisation of the Navier-Stokes Equations on Mixed Grids,” AIAA-1998-0612, 36th AIAA Aerospace Sciences Meeting, January, 1998.

32. Jalai, A., Sharbtdar, M. and Ollivier-Gooch C., “Accuracy analysis of unstructured finite volume discretization schemes,” *Computers & Fluids*, Vol. 101, 2014, pp. 220–232.

33. Wilcox D.C., “Wall matching, a rational alternative to wall functions,” AIAA-89-0611, 27th AIAA Aerospace Sciences Meeting, January, 1989.

34. Wilcox, D.C., **Turbulence Modeling for CFD**, 2nd edition, DCW Industries, Inc., La Canada, CA, 1998.

35. Biedron, R.T., et al., “FUN3D Manual 12.9,” NASA/TM-2016-219012, 2016.

36. Van Driest, E.R., “Turbulent boundary layer in compressible fluids,” *Journal of the Aeronautical Sciences*, Vol. 18, No. 3, 1951, pp. 1012–1028.

37. Bailey, S.C.C., Vallikivi, M. Hultmark, M. and Smits, A.J., “Estimating the value of von Kármán’s constant in turbulent pipe flow,” *Journal of Fluid Mechanics*, Vol. 749, Cambridge University Press, 2014, pp. 79–98.

38. Menter, F.R., "Two-Equation Eddy-Viscosity Turbulence Models for Engineering Applications," *AIAA Journal,* Vol. 32, No. 8, 1994, pp. 1598–160.

39. White, F.M., **Viscous Fluid Flow**, McGraw-Hill, Inc., 1974, pp. 630–646.