Edge-Based Viscous Method for Node-Centered Formulations

Yi Liu[1] and Boris Diskin[2]
National Institute of Aerospace, Hampton, Virginia 23666, USA

William K. Anderson[3], Eric J. Nielsen[4], and Li Wang[5]
NASA Langley Research Center, Hampton, Virginia 23681, USA

This paper presents a novel, efficient, conservative, edge-based method for evaluation of meanflow viscous fluxes and turbulence-model diffusion terms of the Reynolds-averaged Navier-Stokes equations on tetrahedral grids. The new method is implemented in a practical, node-centered, finite-volume computational fluid dynamics solver. The baseline finite-volume scheme that is equivalent to a second-order accurate finite-element Galerkin approximation of viscous stresses is reformulated. The order of operations to compute the cell-based Green-Gauss gradients is changed to combine the operations by edge, which leads to an equivalent formulation on tetrahedral grids, improves efficiency, and preserves the compact discretization stencil based on the nearest neighbors. The computational results presented in this paper verify the implementation of this edge-based method by comparing its accuracy and iterative convergence with those of the well verified and validated baseline formulation. Efficiency gains for residual and Jacobian evaluations result in significant reduction of time to solution. This novel edge-based formulation on tetrahedra can be seamlessly combined with the baseline formulation on cells of other types for computing solutions on mixed-element grids.

I. Introduction

This paper introduces, verifies, and assesses a novel edge-based viscous (EBV) method for node-centered discretizations of elliptic second-order partial differential operators that represent viscous effects in computational fluid dynamics (CFD) equations. The EBV method has recently been implemented in a large-scale CFD code, FUN3D [1], that is developed and maintained at the NASA Langley Research Center (LaRC). FUN3D is widely used for high-fidelity analysis of complex turbulent flows across the speed range from incompressible to hypersonic regimes [2-5]. Standard FUN3D solutions are computed using a spatially second-order accurate, node-centered, finite-volume discretization of the Reynolds-averaged Navier-Stokes (RANS) equations on general unstructured grids. An enhanced hierarchical adaptive nonlinear iteration method (HANIM) [6, 7] has been recently implemented in FUN3D to improve robustness and accelerate convergence of nonlinear iterations. FUN3D provides mature capabilities for multidisciplinary optimization [8] and mesh adaptation [9]. Recent FUN3D porting on advanced computing architectures [10, 11] enables efficient, high-fidelity, scale-resolving simulations of thermochemical nonequilibrium flows for many applications including atmospheric entry, hypersonics, and combustion.

The FUN3D finite-volume (FUN3D-FV) discretization scheme balances conserved fluxes at dual control volumes that are centered at grid points. Inviscid fluxes are evaluated at the edge median in an efficient edge-based loop. Viscous fluxes use the Green-Gauss theorem to compute gradients at grid cells. Because this approach relies on cell-based gradients, the viscous fluxes are implemented in a separate cell-

[1]Senior Research Engineer, Senior Member AIAA
[2]Senior Research Fellow, Associate Fellow AIAA
[3]Senior Research Scientist, Computational Aerosciences Branch, Associate Fellow AIAA
[4]Research Scientist, Computational Aerosciences Branch, Associate Fellow AIAA
[5]Research Aerospace Engineer, Computational Aerosciences Branch, Associate Fellow AIAA

based loop. In this paper, the baseline viscous-flux implementation is referred to as the cell-based viscous (CBV) method. The CBV method provides a compact nearest-neighbor stencil that is suitable for massively parallel computing and conveniently supports an exact linearization. FUN3D-FV solutions have been extensively verified and validated through formal analysis and applications. The EBV method essentially mimics the CBV method on tetrahedral cells but uses an edge-based implementation. The EBV method groups the operations required for computing viscous fluxes by edge. This recombination removes redundant computations inherent in the cell-based flux evaluation loop and significantly reduces the time required for viscous-flux computations. A concession for the improved efficiency is additional memory required to store a few coefficients per edge. The coefficients represent local grid metrics, do not depend on the flow solution, and can be precomputed for static grids. The EBV method maintains the same compact nearest-neighbor stencil for the viscous fluxes, which is a major benefit of the CBV method. The main difference between the CBV method and the EBV method is the viscosity (and other coefficients multiplying gradients in viscous fluxes) evaluation. In the CBV formulation, viscosity is evaluated at the cell centers, while in the EBV formulation, it is evaluated at the edge medians. Both the CBV and EBV methods maintain conservation and second-order accuracy.

In this paper, EBV solutions on tetrahedral grids are verified and demonstrate significant speedup in comparison with the baseline CBV solutions. In this effort, the EBV method has been implemented only for tetrahedra. For mixed-element grids, a hybrid EBV/CBV method is applied: the EBV method is used on tetrahedra and the CBV method is used on cells of other types. The hybrid EBV/CBV solution remains as accurate as the CBV solution on the same grid, but the EBV efficiency benefits are reduced.

The material in the paper is presented in the following order. Section II describes the RANS formulation with a linear one-equation turbulence model used in this study. Section III outlines the discretization methods and the baseline and HANIM iterative solvers available in FUN3D-FV. Section IV describes the details of the EBV method for the viscous fluxes of the meanflow equations and the diffusion term of the turbulence-model equation. Section V compares EBV and CBV solutions for a benchmark flow on a family of tetrahedral grids and analyzes the EBV benefits for the baseline and HANIM iterations. Section VI presents an EBV/CBV solution on a mixed-element grid. Section VII gives concluding remarks. Appendix A discusses the conservation property of the CBV and EBV methods.

II. Reynolds-Averaged Navier-Stokes Equations

The three-dimensional (3D) compressible unsteady RANS equations are given by [12]:

$$\partial_t \boldsymbol{Q} + \partial_x \boldsymbol{F} + \partial_y \boldsymbol{G} + \partial_z \boldsymbol{H} = \boldsymbol{0}. \tag{1}$$

The vectors $\boldsymbol{F}$, $\boldsymbol{G}$, and $\boldsymbol{H}$ are defined as

$$\boldsymbol{F} = \begin{pmatrix} \rho u \\ \rho uu + p - \tau_{xx} \\ \rho uv - \tau_{xy} \\ \rho uw - \tau_{xz} \\ (E+p)u - \left(u\tau_{xx} + v\tau_{xy} + w\tau_{xz}\right) + q_x \end{pmatrix},$$

$$\boldsymbol{G} = \begin{pmatrix} \rho v \\ \rho uv - \tau_{xy} \\ \rho vv + p - \tau_{yy} \\ \rho vw - \tau_{yz} \\ (E+p)v - \left(u\tau_{xy} + v\tau_{yy} + w\tau_{yz}\right) + q_y \end{pmatrix}, \tag{2}$$

$$\boldsymbol{H} = \begin{pmatrix} \rho w \\ \rho uw - \tau_{xz} \\ \rho vw - \tau_{yz} \\ \rho ww + p - \tau_{zz} \\ (E+p)w - \left(u\tau_{xz} + v\tau_{yz} + w\tau_{zz}\right) + q_z \end{pmatrix}.$$

Here, p is the static pressure, $\boldsymbol{u} = (u, v, w)^{\mathrm{T}}$ is the velocity vector, $\boldsymbol{q} = (q_x, q_y, q_z)^{\mathrm{T}}$ is the local heat flux vector, and $\boldsymbol{Q} \equiv (\rho, \rho u, \rho v, \rho w, E)^{\mathrm{T}}$ is the vector of conserved variables that includes the density ρ, the momentum $\rho\boldsymbol{u} = (\rho u, \rho v, \rho w)^{\mathrm{T}}$, and the total energy per unit volume E. The superscript T denotes transposition to indicate vertical vectors. For a perfect gas, equations are closed using the following relations:

$$p = (\gamma - 1)\left(E - \frac{\rho}{2}(u^2 + v^2 + w^2)\right), \qquad a^2 = \gamma\frac{p}{\rho}, \tag{3}$$

where a is the speed of sound and $\gamma = 1.4$ is the ratio of specific heats.

The viscous fluxes in the RANS equations refer to the diffusion terms including the shear stress tensor and the heat flux vector defined as:

$$\begin{aligned}
\tau_{xx} &= \frac{2}{3}\frac{M_{ref}}{Re}(\mu + \mu_t)(2\partial_x u - \partial_y v - \partial_z w),\\
\tau_{yy} &= \frac{2}{3}\frac{M_{ref}}{Re}(\mu + \mu_t)(2\partial_y v - \partial_x u - \partial_z w),\\
\tau_{zz} &= \frac{2}{3}\frac{M_{ref}}{Re}(\mu + \mu_t)(2\partial_z w - \partial_x u - \partial_y v),\\
\tau_{yx} &= \tau_{xy} = \frac{M_{ref}}{Re}(\mu + \mu_t)(\partial_x v + \partial_y u),\\
\tau_{zx} &= \tau_{xz} = \frac{M_{ref}}{Re}(\mu + \mu_t)(\partial_x w + \partial_z u),\\
\tau_{zy} &= \tau_{yz} = \frac{M_{ref}}{Re}(\mu + \mu_t)(\partial_z v + \partial_y w),\\
q_x &= \frac{M_{ref}}{Re(\gamma - 1)}\left(\frac{\mu}{Pr} + \frac{\mu_t}{Pr_t}\right)\partial_x T,\\
q_y &= \frac{M_{ref}}{Re(\gamma - 1)}\left(\frac{\mu}{Pr} + \frac{\mu_t}{Pr_t}\right)\partial_y T,\\
q_z &= \frac{M_{ref}}{Re(\gamma - 1)}\left(\frac{\mu}{Pr} + \frac{\mu_t}{Pr_t}\right)\partial_z T.
\end{aligned} \tag{4}$$

Here, T is the temperature, μ is the dynamic laminar viscosity computed by Sutherland's law [13], μ_t is the turbulent eddy viscosity computed by a turbulence model, M_{ref} is the reference Mach number, Re is the Reynolds number, and Pr and Pr_t are the Prandtl numbers for the meanflow and turbulence models, respectively. In Sutherland's law, the nondimensional local dynamic viscosity, μ, relates to the local non-dimensional temperature, T, through the following formula

$$\mu(T) = T^{\frac{3}{2}}\left(\frac{1 + \frac{S}{T_{ref}}}{T + \frac{S}{T_{ref}}}\right), \tag{5}$$

where $S = 198.6$ºR, and the reference dimensional viscosity, μ_{ref}, is assumed at the reference dimensional temperature T_{ref}.

Following the formulation presented at the NASA Turbulence Modeling Resource (TMR) website*, the standard Spalart-Allmaras (SA) turbulence model [14] is given by the following nonconservative equation:

$$\begin{aligned}
&\partial_t\hat{\nu} + u\partial_x\hat{\nu} + v\partial_y\hat{\nu} + w\partial_z\hat{\nu} - c_{b1}(1 - f_{t2})\hat{S}\hat{\nu} + \left[c_{w1}f_w - \frac{c_{b1}}{\kappa^2}f_{t2}\right]\left(\frac{\hat{\nu}}{d}\right)^2\\
&-\frac{1}{\sigma}\left[\partial_x\big((\nu + \hat{\nu})\partial_x\hat{\nu}\big) + \partial_y\left((\nu + \hat{\nu})\partial_y\hat{\nu}\right) + \partial_z\big((\nu + \hat{\nu})\partial_z\hat{\nu}\big)\right.\\
&\left. + c_{b2}\left((\partial_x\hat{\nu})^2 + \left(\partial_y\hat{\nu}\right)^2 + (\partial_z\hat{\nu})^2\right)\right] = 0.
\end{aligned} \tag{6}$$

* https://turbmodels.larc.nasa.gov/; accessed May 18, 2021

The boundary conditions are the following:

$$\hat{\nu}_{wall} = 0, \qquad \hat{\nu}_{farfield} = 3\nu_{ref}. \tag{7}$$

Here, $\hat{\nu}$ is the turbulence variable, d is the distance to the nearest wall, $\nu = \mu/\rho$ is the kinematic viscosity, and ν_{ref} is the reference kinematic viscosity.

The turbulent eddy viscosity is computed as

$$\mu_t = \max(\rho\hat{\nu}f_{v1}, 0). \tag{8}$$

$$\hat{S} = \Omega + \frac{\hat{\nu}}{\kappa^2 d^2} f_{v2}, \tag{9}$$

where Ω is the magnitude of vorticity,

$$\Omega = \sqrt{\left(\partial_y w - \partial_z v\right)^2 + (\partial_z u - \partial_x w)^2 + \left(\partial_x v - \partial_y u\right)^2}, \tag{10}$$

$$f_{v1} = \frac{\chi^3}{c_{v1}^3 - \chi^3}, \;\; \chi = \frac{\hat{\nu}}{\nu}, \;\; f_{v2} = 1 - \frac{\chi}{1 + \chi f_{v1}}, \;\; f_w = g\left[\frac{1 + c_{w3}^6}{g^6 + c_{w3}^6}\right]^{\frac{1}{6}}, \;\; g = r + c_{w2}(r^6 - r), \tag{11}$$

$$f_{t2} = c_{t3}\exp(-c_{t4}\chi^2), \;\; r = \min\left[\frac{\hat{\nu}}{\hat{S}\kappa^2 d^2}, 10\right], \tag{12}$$

and the constants are $\kappa = 0.41$, $\sigma = \frac{2}{3}$, $c_{b1} = 0.1355$, $c_{b2} = 0.622$, $c_{w1} = \frac{c_{b1}}{\kappa} + \frac{1+c_{b2}}{\sigma}$, $c_{w2} = 0.3$, $c_{w3} = 2$, $c_{v1} = 7.1$, $c_{t3} = 1.2$, and $c_{t4} = 0.5$.

The standard SA model equation is solved for $\hat{\nu} > 0$. For negative $\hat{\nu}$, the following equation [15] is solved

$$\begin{aligned} &\partial_t\hat{\nu} + u\partial_x\hat{\nu} + v\partial_y\hat{\nu} + w\partial_z\hat{\nu} - c_{b1}(1 - c_{t3})\Omega\hat{\nu} - c_{w1}\left(\frac{\hat{\nu}}{d}\right)^2 \\ &- \frac{1}{\sigma}\Big[\partial_x\big((\nu + \hat{\nu}f_n)\partial_x\hat{\nu}\big) + \partial_y\left((\nu + \hat{\nu}f_n)\partial_y\hat{\nu}\right) + \partial_z\big((\nu + \hat{\nu}f_n)\partial_z\hat{\nu}\big) \\ &+ c_{b2}\left((\partial_x\hat{\nu})^2 + \left(\partial_y\hat{\nu}\right)^2 + (\partial_z\hat{\nu})^2\right)\Big] = 0. \end{aligned} \tag{13}$$

with

$$f_n = \frac{c_{n1} + \chi^3}{c_{n1} - \chi^3}, \qquad c_{n1} = 16. \tag{14}$$

The SA model extended to negative values of $\hat{\nu}$ is referenced as the SA-neg model. The SA-neg turbulent eddy viscosity is also computed using Eq. 8.

III. Baseline Discretization and Iteration Methods

This section overviews the baseline FUN3D-FV discretization method, including the CBV method for the meanflow inviscid fluxed and the turbulence-model diffusion term. The baseline and HANIM iterative solvers are also overviewed.

A. FUN3D-FV Discretization Scheme

FUN3D is a general-purpose solver for the RANS equations discretized on unstructured mixed-element grids that may contain tetrahedra, pyramids, prisms, and hexahedra. The FUN3D-FV residuals are evaluated on a set of median-dual control volumes centered around grid points. Edge-based inviscid fluxes are computed at primal edge medians using an approximate Riemann solver. In the current study, Roe's flux difference splitting [16] is used. For second-order accuracy, density, pressure, and velocity are reconstructed by a UMUSCL (Unstructured Monotonic Upstream-centered Scheme for Conservation Laws) scheme [17, 18]. The spatial discretization of the SA-neg turbulence model uses a first-order accurate convection scheme. For the discretization of viscous fluxes, the Green-Gauss theorem is used to compute cell-based gradients. On tetrahedral meshes, this CBV approach is equivalent to a Galerkin approximation [19]. For nontetrahedral meshes, cell-based Green-Gauss gradients are combined with edge-based gradients [20-22] to improve stability of viscous operators and prevent odd-even decoupling. The diffusion term in

the turbulence model is handled similarly, with the exception that the edge-based gradient augmentation is performed at all cells, including tetrahedra. The vorticity-based source term for the turbulence model is computed using velocity gradients evaluated by the Green-Gauss method on dual control volumes. The boundary conditions involved in the present study include farfield Riemann-invariant, farfield Roe-based, symmetry, and viscous-wall boundary conditions [23].

B. Baseline Iterative Solver

The general form of the unsteady RANS equations is given by Eq. 1. For steady-state computations, the time derivative can be dropped, leading to a formal equation

$$\boldsymbol{R}(\boldsymbol{Q}) = 0, \tag{15}$$

where $\boldsymbol{Q}$ denotes the flow solution and $\boldsymbol{R}$ denotes the discrete nonlinear steady-state residual. This nonlinear system of equations is solved by a defect-correction method

$$\boldsymbol{D}\Delta\boldsymbol{Q} + \frac{\boldsymbol{\partial}\widehat{\boldsymbol{R}}}{\boldsymbol{\partial Q}}\,\Delta\boldsymbol{Q} = -\boldsymbol{R}(\boldsymbol{Q}^n), \tag{16}$$

$$\boldsymbol{Q}^{n+1} = \boldsymbol{Q}^n + \Delta\boldsymbol{Q}. \tag{17}$$

Here, $\boldsymbol{Q}^n$ and $\boldsymbol{Q}^{n+1}$ are the solutions at iterations n and $n+1$, respectively. The term $\frac{\boldsymbol{\partial}\widehat{\boldsymbol{R}}}{\boldsymbol{\partial Q}}$ approximates the Jacobian $\frac{\boldsymbol{\partial R}}{\boldsymbol{\partial Q}}$, $\boldsymbol{D}$ is a diagonal matrix with $\frac{V}{\Delta\tau}$ on the diagonal, V is a control volume, and $\Delta\tau$ is a pseudotime step, which is set through a Courant-Friedrichs-Lewy (CFL) number specification. The meanflow and SA-neg equations are loosely coupled. The approximate Jacobian for the meanflow equations is formed using the linearization of the first-order flux-vector splitting inviscid fluxes [24] and the second-order viscous fluxes.

The approximate Jacobian for the SA-neg equation includes the contributions from the advection, diffusion, and source terms. The advection term is linearized with a first-order approximation. The exact linearization is used for the diffusion term. The entire contribution from the linearized source term is added to the diagonal. The absolute value of the diagonal term is placed on the diagonal of the turbulence-model equation to enforce positivity prior to adding the pseudotime term. Nonlinear iterations can reuse the Jacobian computed at a previous iteration; Jacobian updates are scheduled according to residual convergence.

At each nonlinear iteration, the linear system represented by Eq. 16 is solved using a specified number of point-implicit Gauss-Seidel (GS) sweeps with multicolor ordering. The resulting $\Delta\boldsymbol{Q}$ is added to the nonlinear solution. The CFL number can be predefined or ramped linearly within a specified number of nonlinear iterations.

C. HANIM Iterative Solver

HANIM is a strong nonlinear solver that is based on a hierarchy of modules including a preconditioner, a Newton-Krylov linear solver, realizability check, nonlinear control, and CFL adaption modules. The specific HANIM implementation [7] follows Refs. [6, 25, 26]. The goal of HANIM is to enhance the iterative scheme with a mechanism for an automatic adaption of the pseudotime step to increase convergence rate and overcome instabilities occurring in transient solutions. The preconditioner is similar to the baseline defect-correction method. The matrix-free linear solver [27-31] uses Fréchet derivatives and a Generalized Conjugate Residual (GCR) [28] method from the family of Krylov methods. HANIM prescribes the residual reduction targets for the preconditioner, GCR, and nonlinear solution updates and specifies the maximum number of linear iterations allowed in the preconditioner and the maximum number of search directions allowed in GCR. Unlike the baseline iterative solver, the HANIM CFL update strategy is adaptive. HANIM increases the CFL number if all the HANIM modules have reported success. On the other hand, if any of the modules fail, HANIM discards the suggested correction and aggressively reduces the CFL.

IV. Edge-Based Viscous (EBV) Method for Tetrahedral Cells

Node-centered edge-based finite-volume schemes for inviscid fluxes are widely used in unstructured-grid solvers, including FUN3D-FV. Edge-based schemes on tetrahedral grids offer advantages of efficiency and generality. In addition, third-order accuracy for the inviscid fluxes can be achieved [32-35] on tetrahedral grids.

Edge-based schemes for viscous fluxes have been proposed in the literature as well [36-39]. These schemes typically require extended stencils that include neighbors of neighbors. The EBV method proposed in this paper is derived from the baseline CBV method and preserves the compact CBV stencil that includes only immediate neighbors. The CBV gradient at a cell is computed using a Green-Gauss approach. For the tetrahedron in Fig. 1, vector $\boldsymbol{n}_i = \left(n_{ix}, n_{iy}, n_{iz}\right)^{\mathrm{T}}, i = 1, 2, 3,$ and 4, is the outward directed area of the triangular face opposite to the point $\boldsymbol{p}_i$. For a discrete function φ defined at grid points, its gradient can be computed as

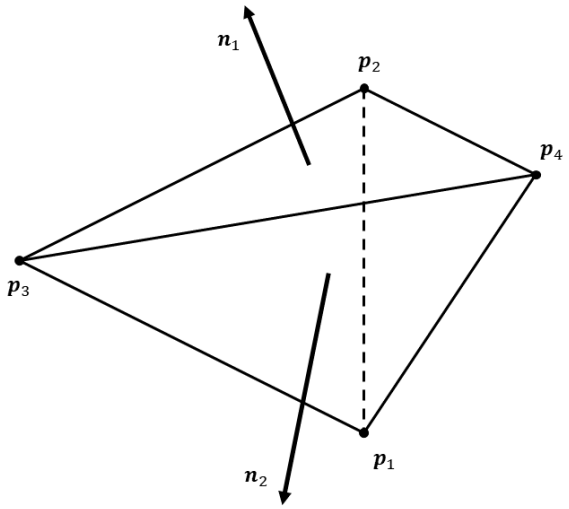


Figure 1. Tetrahedron sketch.

$$\boldsymbol{\nabla}\varphi = \frac{(\varphi_2 + \varphi_3 + \varphi_4)\boldsymbol{n}_1 + (\varphi_1 + \varphi_3 + \varphi_4)\boldsymbol{n}_2 + (\varphi_1 + \varphi_2 + \varphi_4)\boldsymbol{n}_3 + (\varphi_1 + \varphi_2 + \varphi_3)\boldsymbol{n}_4}{3\,Vol}. \quad (18)$$

Here, Vol is the volume of the tetrahedron, and $\boldsymbol{\nabla} \equiv \left(\partial_x, \partial_y, \partial_z\right)^{\mathrm{T}}$ denotes a formal vector differentiation operator. For a closed tetrahedron,

$$\boldsymbol{n}_1 + \boldsymbol{n}_2 + \boldsymbol{n}_3 + \boldsymbol{n}_4 = 0. \quad (19)$$

Introducing the edge-based difference operator

$$\Delta_{ij}\varphi \equiv \varphi_i - \varphi_j. \quad (20)$$

and substituting $\boldsymbol{n}_1 = -(\boldsymbol{n}_2 + \boldsymbol{n}_3 + \boldsymbol{n}_4)$ in Eq. 18, one can obtain

$$\boldsymbol{\nabla}\varphi = \frac{-1}{3\,Vol}(\Delta_{21}\varphi\boldsymbol{n}_2 + \Delta_{31}\varphi\boldsymbol{n}_3 + \Delta_{41}\varphi\boldsymbol{n}_4). \quad (21a)$$

Analogously,

$$\boldsymbol{\nabla}\varphi = \frac{-1}{3\,Vol}(\Delta_{12}\varphi\boldsymbol{n}_1 + \Delta_{32}\varphi\boldsymbol{n}_3 + \Delta_{42}\varphi\boldsymbol{n}_4), \quad (21b)$$

$$\boldsymbol{\nabla}\varphi = \frac{-1}{3\,Vol}(\Delta_{13}\varphi\boldsymbol{n}_1 + \Delta_{23}\varphi\boldsymbol{n}_2 + \Delta_{43}\varphi\boldsymbol{n}_4), \quad (21c)$$

$$\boldsymbol{\nabla}\varphi = \frac{-1}{3\,Vol}(\Delta_{14}\varphi\boldsymbol{n}_1 + \Delta_{24}\varphi\boldsymbol{n}_2 + \Delta_{34}\varphi\boldsymbol{n}_3). \quad (21d)$$

Equations 21a-21d express the cell-based gradient (Eq. 18) in terms of the edge-based differences.

The same approach can be extended to the viscous terms of the RANS equations. Recall that only four meanflow equations have contributions from viscous fluxes; the mass conservation equation does not include viscous fluxes. The viscous shear stress contribution from a cell to the x-, y-, and z-momentum conservation residuals evaluated at the grid point $\boldsymbol{p}_i$ can be written as:

$$R_{mxi} = R_{mxi} - \frac{M_{ref}}{Re}\frac{\mu+\mu_t}{V_i}\left[-\frac{2}{3}\left(\boldsymbol{\nabla}^h\cdot\boldsymbol{u}\right)d_{ix} + \left(\boldsymbol{d}_i\cdot\boldsymbol{\nabla}^h\right)u + \partial_x^h(\boldsymbol{d}_i\cdot\boldsymbol{u})\right], \tag{22}$$

$$R_{myi} = R_{myi} - \frac{M_{ref}}{Re}\frac{\mu+\mu_t}{V_i}\left[-\frac{2}{3}\left(\boldsymbol{\nabla}^h\cdot\boldsymbol{u}\right)d_{iy} + \left(\boldsymbol{d}_i\cdot\boldsymbol{\nabla}^h\right)v + \partial_y^h(\boldsymbol{d}_i\cdot\boldsymbol{u})\right], \tag{23}$$

$$R_{mzi} = R_{mzi} - \frac{M_{ref}}{Re}\frac{\mu+\mu_t}{V_i}\left[-\frac{2}{3}\left(\boldsymbol{\nabla}^h\cdot\boldsymbol{u}\right)d_{iz} + \left(\boldsymbol{d}_i\cdot\boldsymbol{\nabla}^h\right)w + \partial_z^h(\boldsymbol{d}_i\cdot\boldsymbol{u})\right]. \tag{24}$$

The residuals in Eqs. 22-24 are divided by the control volume V_i associated with the grid point $\boldsymbol{p}_i$ to approximate the differential formulation of the momentum conservation equations (Eq. 2). The corresponding undivided residual evaluating the balance of discrete fluxes can be written in a vector form as

$$\boldsymbol{R}_{mi} = \boldsymbol{R}_{mi} - (\mu+\mu_t)\left[-\frac{2}{3}\left(\boldsymbol{\nabla}^h\cdot\boldsymbol{u}\right)\boldsymbol{d}_i + \left(\boldsymbol{d}_i\cdot\boldsymbol{\nabla}^h\right)\boldsymbol{u} + \boldsymbol{\nabla}^h(\boldsymbol{d}_i\cdot\boldsymbol{u})\right]. \tag{25}$$

Here, $\boldsymbol{R}_{mi} = \frac{Re}{M_{ref}}V_i\left(R_{mxi}, R_{myi}, R_{mzi}\right)^{\mathrm{T}}$ is the vector of momentum-conservation residuals, $\boldsymbol{d}_i = \left(d_{ix}, d_{iy}, d_{iz}\right)^{\mathrm{T}}$ is a directed area vector [2] of the control-volume boundary located within the cell, and the superscript h denotes discretization of the cell gradient. For a tetrahedron, the directed area of the control-volume boundary associated with point $\boldsymbol{p}_i$ relates to the directed area of the opposite face as

$$\boldsymbol{d}_i = \frac{1}{3}\boldsymbol{n}_i. \tag{26}$$

For the momentum conservation residual (Eq. 25) evaluated at grid point $\boldsymbol{p}_1$ of the tetrahedron shown in Fig. 1, the cell gradients can be expressed in terms of the edge-based differences corresponding to edges that include cell point $\boldsymbol{p}_1$ as in Eq. 21a. Then

$$\boldsymbol{\nabla}^h\varphi = \frac{-1}{3\,Vol}\left(\Delta_{21}\varphi\boldsymbol{n}_2 + \Delta_{31}\varphi\boldsymbol{n}_3 + \Delta_{41}\varphi\boldsymbol{n}_4\right), \tag{27}$$

$$\begin{aligned}\boldsymbol{R}_{m1} = \boldsymbol{R}_{m1} + \frac{\mu+\mu_t}{3\,Vol}\Big[&-\frac{2}{3}\left((\boldsymbol{n}_2\cdot\Delta_{21}\boldsymbol{u})\boldsymbol{d}_1 + (\boldsymbol{n}_3\cdot\Delta_{31}\boldsymbol{u})\boldsymbol{d}_1 + (\boldsymbol{n}_4\cdot\Delta_{41}\boldsymbol{u})\boldsymbol{d}_1\right)\\ &+\left((\boldsymbol{n}_2\cdot\boldsymbol{d}_1)\Delta_{21}\boldsymbol{u} + (\boldsymbol{n}_3\cdot\boldsymbol{d}_1)\Delta_{31}\boldsymbol{u} + (\boldsymbol{n}_4\cdot\boldsymbol{d}_1)\Delta_{41}\boldsymbol{u}\right)\\ &+\left((\boldsymbol{d}_1\cdot\Delta_{21}\boldsymbol{u})\boldsymbol{n}_2 + (\boldsymbol{d}_1\cdot\Delta_{31}\boldsymbol{u})\boldsymbol{n}_3 + (\boldsymbol{d}_1\cdot\Delta_{41}\boldsymbol{u})\boldsymbol{n}_4\right)\Big],\end{aligned} \tag{28}$$

where $\Delta_{ji}\boldsymbol{u} = \left(\Delta_{ji}u, \Delta_{ji}v, \Delta_{ji}w\right)^{\mathrm{T}}$, $\Delta_{ji}u = u_j - u_i$, $\Delta_{ji}v = v_j - v_i$, $\Delta_{ji}w = w_j - w_i$.

Equation 28 is equivalent to the CBV method. The terms in square brackets are edge-based terms. The viscosity coefficient appearing in front of the square bracket is the only cell-based term. To enable an edge-based implementation, the viscosity must instead be evaluated along the edge. The EBV method modifies Eq. 28 as

$$\begin{aligned}\boldsymbol{R}_{m1} = \boldsymbol{R}_{m1} + \frac{1}{3\,Vol}\Bigg[&(\mu+\mu_t)_{21}\left((\boldsymbol{n}_2\cdot\boldsymbol{d}_1)\Delta_{21}\boldsymbol{u} - \frac{2}{3}(\boldsymbol{n}_2\cdot\Delta_{21}\boldsymbol{u})\boldsymbol{d}_1 + (\boldsymbol{d}_1\cdot\Delta_{21}\boldsymbol{u})\boldsymbol{n}_2\right)\\ &+(\mu+\mu_t)_{31}\left((\boldsymbol{n}_3\cdot\boldsymbol{d}_1)\Delta_{31}\boldsymbol{u} - \frac{2}{3}(\boldsymbol{n}_3\cdot\Delta_{31}\boldsymbol{u})\boldsymbol{d}_1 + (\boldsymbol{d}_1\cdot\Delta_{31}\boldsymbol{u})\boldsymbol{n}_3\right)\\ &+(\mu+\mu_t)_{41}\left((\boldsymbol{n}_4\cdot\boldsymbol{d}_1)\Delta_{41}\boldsymbol{u} - \frac{2}{3}(\boldsymbol{n}_4\cdot\Delta_{41}\boldsymbol{u})\boldsymbol{d}_1 + (\boldsymbol{d}_1\cdot\Delta_{41}\boldsymbol{u})\boldsymbol{n}_4\right)\Bigg],\end{aligned} \tag{29}$$

where

$$(\mu+\mu_t)_{ji} = \frac{(\mu+\mu_t)_j + (\mu+\mu_t)_i}{2}. \tag{30}$$

represents the edge-based average of the viscosity coefficients defined at points $\boldsymbol{p}_i$ and $\boldsymbol{p}_j$.

Substituting Eq. 26 into Eq. 29, the contribution from edge $[\boldsymbol{p}_1, \boldsymbol{p}_2]$ to the momentum-conservation residual evaluated at grid point $\boldsymbol{p}_1$ can be represented as

$$\begin{aligned} \boldsymbol{R}_{m1} &= \boldsymbol{R}_{m1} + \frac{(\mu+\mu_t)_{21}}{9\,Vol}\left[(\boldsymbol{n}_2\cdot\boldsymbol{n}_1)\Delta_{21}\boldsymbol{u} - \frac{2}{3}(\boldsymbol{n}_2\cdot\Delta_{21}\boldsymbol{u})\boldsymbol{n}_1 + (\boldsymbol{n}_1\cdot\Delta_{21}\boldsymbol{u})\boldsymbol{n}_2\right] \\ &= \boldsymbol{R}_{m1} + \frac{(\mu+\mu_t)_{21}}{9\,Vol}\left[(\boldsymbol{n}_2\cdot\boldsymbol{n}_1)\boldsymbol{I} - \frac{2}{3}\boldsymbol{n}_1\boldsymbol{n}_2^{\mathrm{T}} + \boldsymbol{n}_2\boldsymbol{n}_1^{\mathrm{T}}\right]\Delta_{21}\boldsymbol{u}. \end{aligned} \tag{31}$$

Here, $\boldsymbol{I}$ is the 3×3 identity matrix and the 3×3 matrix in the square brackets represent the nine EBV coefficients. The EBV coefficients are summed over all tetrahedra that share edge $[\boldsymbol{p}_1, \boldsymbol{p}_2]$. The resulting nine EBV coefficients represent the contribution from the edge to the momentum-conservation residual at the point $\boldsymbol{p}_1$. In general, $\boldsymbol{n}_1\boldsymbol{n}_2^T \neq \boldsymbol{n}_2\boldsymbol{n}_1^T$, implying that some edge contributions to the momentum-conservation residuals from individual cells are not symmetric with respect to the edge endpoints. This lack of symmetry indicates that the EBV method does not conserve edge-based contributions from individual cells. However, as shown in the appendix, the matrix of the EBV coefficients collected over all tetrahedra that surround and share the edge is expected to be symmetric with respect to the edge endpoints, indicating the global conservation property.

In the initial EBV implementation for the meanflow viscous fluxes, a separate set of coefficients has been allocated for each endpoint of the edge. This approach doubles the memory requirements. Recently, it has been recognized that, the two 3×3 matrices, $\left[(\boldsymbol{n}_2\cdot\boldsymbol{n}_1)\boldsymbol{I} - \frac{2}{3}\boldsymbol{n}_1\boldsymbol{n}_2^{\mathrm{T}} + \boldsymbol{n}_2\boldsymbol{n}_1^{\mathrm{T}}\right]$ and $\left[(\boldsymbol{n}_1\cdot\boldsymbol{n}_2)\boldsymbol{I} - \frac{2}{3}\boldsymbol{n}_2\boldsymbol{n}_1^{\mathrm{T}} + \boldsymbol{n}_1\boldsymbol{n}_2^{\mathrm{T}}\right]$, of the momentum-residual EBV coefficients (Eq. 31) that correspond to the two edge endpoints are transpose of each other. Consequently, it is sufficient to store only one set of the EBV coefficients. Moreover, as discussed in Appendix, for the interior edges surrounded by tetrahedra, the matrices are expected to be symmetric. Thus, there are only six independent coefficients for the momentum-conservation residual.

For the energy conservation equation, the term with the shear stress tensor has the same EBV formulation as in the momentum equations. The same EBV coefficients are used; no additional computations and storage are needed. However, the heat flux,

$$\boldsymbol{q} = \frac{M_{ref}}{Re(\gamma-1)}\left(\frac{\mu}{Pr} + \frac{\mu_t}{Pr_t}\right)\boldsymbol{\nabla}T\,, \tag{32}$$

contribution to the energy equation residual needs to be reformulated using the edge-based differences, and one additional EBV coefficient is needed.

$$R_{e1} = R_{e1} + \frac{\left(\frac{\mu}{Pr} + \frac{\mu_t}{Pr_t}Pr_t\right)_{21}}{9\,(\gamma-1)Vol}(\boldsymbol{n}_2\cdot\boldsymbol{n}_1)\Delta_{21}T, \tag{33}$$

where

$$\left(\frac{\mu}{Pr} + \frac{\mu_t}{Pr_t}\right)_{ji} = \frac{1}{2}\left(\left(\frac{\mu}{Pr} + \frac{\mu_t}{Pr_t}\right)_j + \left(\frac{\mu}{Pr} + \frac{\mu_t}{Pr_t}\right)_i\right). \tag{34}$$

Overall, the optimized EBV implementation of the momentum and energy conservation equations requires storage of seven coefficients for each edge: six coefficients to compute the shear stresses and one coefficient to compute the heat flux. These EBV coefficients depend only on grid metrics and can be precomputed for static-grid simulations.

The EBV method has also been implemented for the diffusion term of the SA-neg turbulence model. The nonlinear diffusion term of Eq. 13 is the term in the square brackets. Direct discretization of the term $(\boldsymbol{\nabla}\tilde{\nu}\cdot\boldsymbol{\nabla}\tilde{\nu})$ associated with the coefficient c_{b2} is not straightforward and may negatively affect stability of nonlinear iterations. Contributions from this term to the linearization matrix have an unpredictable and uncontrollable effect because of lack of diagonal dominance and variation of the sign of $\tilde{\nu}$ derivatives. These considerations are especially important for iterative methods that use simple point- and line-implicit iterations, which invert the diagonal blocks.

To facilitate discretization, the following transformation is performed:

$$(\boldsymbol{\nabla}\tilde{\nu}\cdot\boldsymbol{\nabla}\tilde{\nu}) = \boldsymbol{\nabla}\cdot(\tilde{\nu}\boldsymbol{\nabla}\tilde{\nu}) - \tilde{\nu}(\boldsymbol{\nabla}\cdot\boldsymbol{\nabla})\tilde{\nu}. \tag{35}$$

This transformation is exact for the differential operators. The first term, $\boldsymbol{\nabla}\cdot(\tilde{\nu}\boldsymbol{\nabla}\tilde{\nu})$, is a conservative nonlinear diffusion operator that can be discretized by a finite-volume method. The second term, $\tilde{\nu}(\boldsymbol{\nabla}\cdot\boldsymbol{\nabla})\tilde{\nu}$, is the Laplace operator multiplied by the local solution. The Laplace operator is a particular example of a linear diffusion operator that can also be discretized by a finite-volume method. The discretization is non-conservative because the Laplacian is multiplied by the local solution.

The modified nonlinear diffusion term can be reformulated as

$$\left[\boldsymbol{\nabla}\cdot\left((\nu+\tilde{\nu}f_n)\boldsymbol{\nabla}\tilde{\nu}\right) + c_{b2}(\boldsymbol{\nabla}\tilde{\nu}\cdot\boldsymbol{\nabla}\tilde{\nu})\right] = \left[\boldsymbol{\nabla}\cdot\left((\nu+\tilde{\nu}f_n + c_{b2}\tilde{\nu})\boldsymbol{\nabla}\tilde{\nu}\right) - c_{b2}\tilde{\nu}(\boldsymbol{\nabla}\cdot\boldsymbol{\nabla})\tilde{\nu}\right]. \tag{36}$$

The contribution from a control-volume face to the residual at grid point $\boldsymbol{p}_i$ can be expressed as

$$\left[(\nu+\tilde{\nu}f_n + c_{b2}\tilde{\nu})_{face} - c_{b2}\tilde{\nu}_i\right]\boldsymbol{\nabla}\tilde{\nu}. \tag{37}$$

Here, the subscript $face$ indicates the quantity evaluated either at the cell centroid for the CBV method or at the edge median for the EBV method. The subscript i indicates the solution at the grid point where the residual is evaluated, which is either a cell vertex for the CBV method or an edge endpoint for the EBV method. Since computing the SA-neg diffusion flux requires the same metrics as those used to compute the heat flux in Eq. 33, there is no need to compute and store additional EBV coefficients for the SA-neg diffusion term.

Currently, the EBV method has been implemented only for tetrahedra. On mixed-element grids, the EBV method for tetrahedra is seamlessly combined with the CBV method for cells of other types. The EBV coefficients are stored for all edges that belong to a tetrahedral cell. The viscous fluxes corresponding to tetrahedral cells are computed in an edge loop. The viscous fluxes corresponding to cells of other types are computed in a separate loop over cells. Jacobian evaluation uses a similar approach.

V. EBV Solutions on Tetrahedral Grids

This section compares the CBV and EBV methods and the corresponding time to solution on tetrahedral grids. Fully converged solutions are computed for an established benchmark turbulent flow. Performance of the baseline and HANIM iterative solvers is discussed. Residual and Jacobian components associated with the CBV and EBV methods are profiled.

Benchmark flow conditions and tetrahedral grid family

The three-dimensional benchmark flow considered in this section is a subsonic separated flow around a hemisphere-cylinder configuration [40]. The cylinder and hemisphere have diameters of unity. The combined length of the configuration is 10. The apex of the hemisphere is located at the origin of the coordinate system. The cylinder axis is aligned with the x-axis. The outflow conditions are assigned at a plane that is orthogonal to the x-axis and contains the cylinder base located at $x = 10$. The symmetry condition is assigned at the vertical plane corresponding to $y = 0$. The farfield boundary is a quadrant of a sphere $((x-10)^2 + y^2 + z^2 = r^2,\ (10-r) \le x \le 10,\ 0 \le y \le r,\ -r \le z \le r)$ with the radius $r = 100$. The flow corresponds to the reference (freestream) Mach number of 0.6, the Reynolds number of 3.5×10^5 based on the unit length, the angle of attack of 19°, and the reference temperature of 540 °R.

A family of three nested tetrahedral grids has been generated using the FORTRAN programs available at the TMR website. The fine T1 grid has 8,995,153 grid points, 53,084,160 tetrahedra, and the surface triangulation composed of 27,648 faces. The T2 and T3 grids are derived from the T1 grid using the grid-coarsening program also available at the TMR website. Table 1 provides the grid statistics. Figure 2 shows the volume and surface meshes corresponding to the T3 grid, where red color indicates the cylinder surface, blue color shows the symmetry boundary, green color shows the outflow boundary and orange color marks the farfield boundary.

Table 1. Family of tetrahedral grids for hemisphere cylinder.

Grid	Points	Cells	Edges
T1	8,995,153	53,084,160	62,373,200
T2	1,143,081	6,635,520	7,852,072
T3	147,637	829,440	995,444

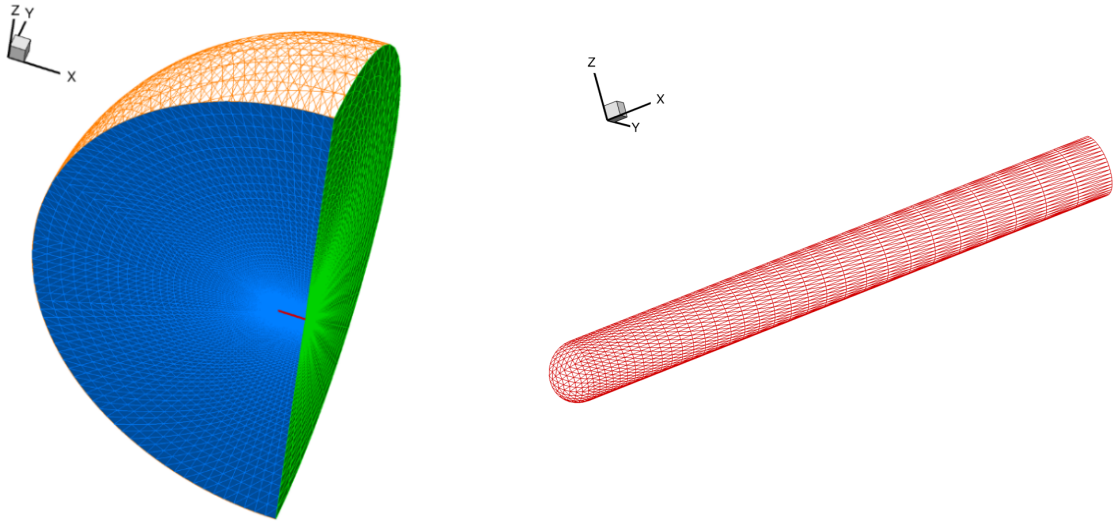


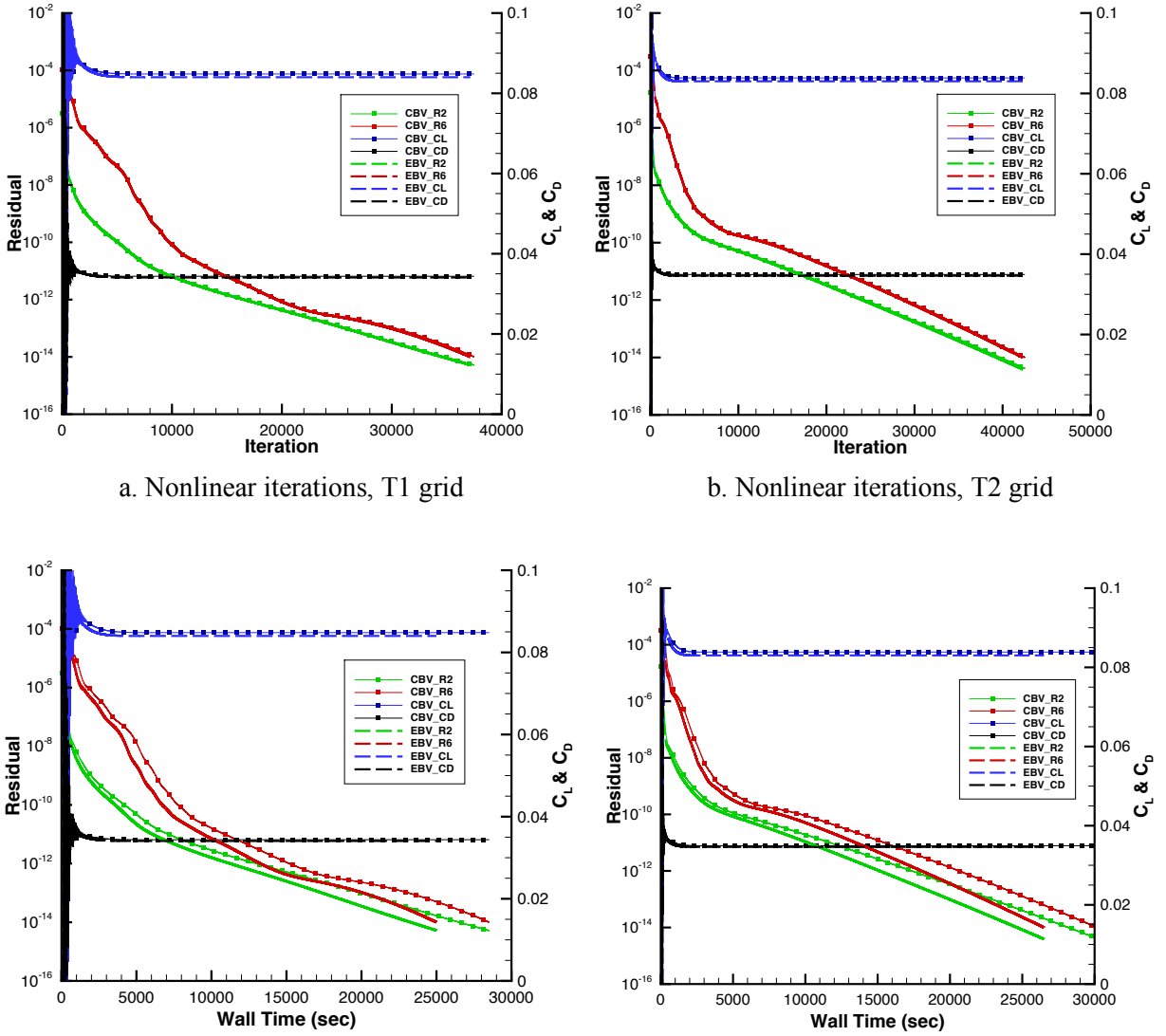
Figure 2. Volume and surface mesh for benchmark flow around hemisphere cylinder.

Baseline iterations

To verify the EBV implementation, convergence of the baseline iterations for the EBV and CBV methods is compared on the family of tetrahedral grids. The EBV solutions computed in this section are not optimized for memory requirements and use 20 EBV coefficients per edge. The iteration stopping criterion is set as 10^{-14} for the root-mean-square (rms) norm of residuals. The CFL number is set to 100. The decoupled linear system is solved using 30 multicolor sweeps for the meanflow and 15 multicolor sweeps for the SA-neg equation. The UMUSCL parameter is set to $\kappa = 0.75$. Figure 3 shows convergence of the CBV and SBV methods on the T2 and T1 grids. R2 and R6 denote the rms norm of the x-momentum and SA-neg residuals, respectively; C_D and C_L denote the drag and lift coefficients. As expected, the EBV and CBV methods show almost identical convergence per iteration and a close agreement between the converged aerodynamic coefficients. The EBV and CBV convergence plots shown in Figs. 3a and 3b are hardly distinguishable. The converged lift and drag coefficients computed on the three tetrahedral grids differ by less than 1% as shown in Table 2. The iterative convergence history versus wall time in seconds is shown in Figs. 3c and 3d. The EBV method takes less time per iteration than the CBV method and significantly reduces the time to convergence.

Table 2. Aerodynamic coefficients for hemisphere-cylinder benchmark flow.

Grid	Lift Coefficient		Drag Coefficient	
	CBV	EBV	CBV	EBV
T1	**0.084**8187	**0.084**0070	**0.034**3821	**0.034**0752
T2	**0.083**8571	**0.083**0166	**0.034**9403	**0.034**6277
T3	**0.08**45361	**0.08**37683	**0.038**3544	**0.038**0831



a. Nonlinear iterations, T1 grid

b. Nonlinear iterations, T2 grid

c. Wall time, T1 grid

d. Wall time, T2 grid

Figure 3. Baseline iterations on tetrahedral grids.

The overall performance of the EBV and CBV methods is quantitatively assessed in Table 3. For each grid, the number of Central Processing Unit (CPU) cores and the wall time used to reach the converged solution from the freestream setup are shown. The EBV speedup is defined as the difference between the CBV time and the EBV time divided by the CBV time and multiplied by 100%. The solutions are computed on the NASA LaRC K4 cluster using Intel Gold 6148 Skylake compute nodes. Each node has a dual socket of 20 processing cores per socket. Intel® Fortran 2019 compiler has been used with the optimization level "–O2", which is standard for FUN3D production runs. On all grids, each individual partition has approximately 30,000 grid points. The solver parameters are the same as in the verification study described above. The EBV speedup is at least 12% and appears to be independent of the grid size.

Table 3. Overall performance on tetrahedral grids.

Grid	CPU cores	Wall time to solution		
		CBV (sec)	EBV (sec)	EBV speedup
T1	320	28,478	24,956	12%
T2	40	30,174	26,502	12%
T3	5	11,685	9,946	15%

Tables 4 and 5 compare the wall time used by the CBV and EBV methods to compute viscous fluxes and Jacobians for a single nonlinear iteration. The timing for such computations has been averaged over five iterations and recorded separately for the meanflow and SA-neg equations. The EBV method shows at least 35% speedup in computing the viscous-flux contributions to the meanflow residuals, 85% speedup in computing the diffusion term contribution to the SA-neg residual, over 39% speedup in computing the viscous-flux contributions to the meanflow Jacobian, and 92% speedup in computing the diffusion contributions to the SA-neg Jacobian.

Table 4. Time for contributions to residual on tetrahedral grids (sec).

Grid	Meanflow viscous fluxes			SA-neg diffusion		
	CBV	EBV	EBV speedup	CBV	EBV	EBV speedup
T1	0.020	0.013	35%	0.065	0.010	85%
T2	0.020	0.012	37%	0.061	0.009	85%
T3	0.014	0.009	38%	0.042	0.006	85%

Table 5. Time for contributions to Jacobian on tetrahedral grids (sec).

Grid	Meanflow viscous fluxes			SA-neg diffusion		
	CBV	EBV	EBV speedup	CBV	EBV	EBV speedup
T1	0.132	0.079	40%	0.160	0.012	92%
T2	0.131	0.076	42%	0.158	0.012	92%
T3	0.084	0.051	39%	0.104	0.008	92%

In practice, the frequency of the Jacobian updates depends on nonlinear residual convergence. Many nonlinear iterations reduce their computational cost by avoiding Jacobian evaluation. The EBV speedup of an individual nonlinear iteration depends on the fraction of time that is spent on operations related to viscous terms. This fraction is significantly higher when Jacobians are evaluated. Comparison of CBV and EBV timing for one nonlinear iteration that performs Jacobian evaluation is shown in Table 6. The CBV computations take more than 32% of the corresponding nonlinear iteration; the EBV computations take less than 15% of the corresponding nonlinear iteration. The EBV method speeds up computations related to viscous terms by 70% (time reduction by more than a factor of 3) on all grids. The overall speedup is more than 23%. Table 7 provides the data for a nonlinear iteration that does not update the Jacobians. The fraction of the CBV computations is reduced to less than 15%; the EBV computations take less than 4% of the corresponding nonlinear iteration. The EBV method speeds up the computations of the meanflow viscous fluxes and SA-neg diffusion terms by 73% but because of a small fraction of time that such nonlinear iteration spends on viscous computations, the overall speedup is about 8%. The EBV speedup for the baseline iterative solver on tetrahedral grids is expected to be between 8% and 23%, depending on the fraction of nonlinear iterations that perform Jacobian evaluation.

Table 6. Time for nonlinear iteration with Jacobian evaluation (sec).

Grid	Meanflow viscous fluxes and SA-neg diffusion: residuals and Jacobian			Complete nonlinear iteration		
	CBV	EBV	EBV speedup	CBV	EBV	EBV speedup
T1	0.378	0.114	70%	1.176	0.909	23%
T2	0.370	0.110	70%	1.115	0.851	24%
T3	0.245	0.074	70%	0.663	0.495	25%

Table 7. Time for nonlinear iteration without Jacobian evaluation (sec).

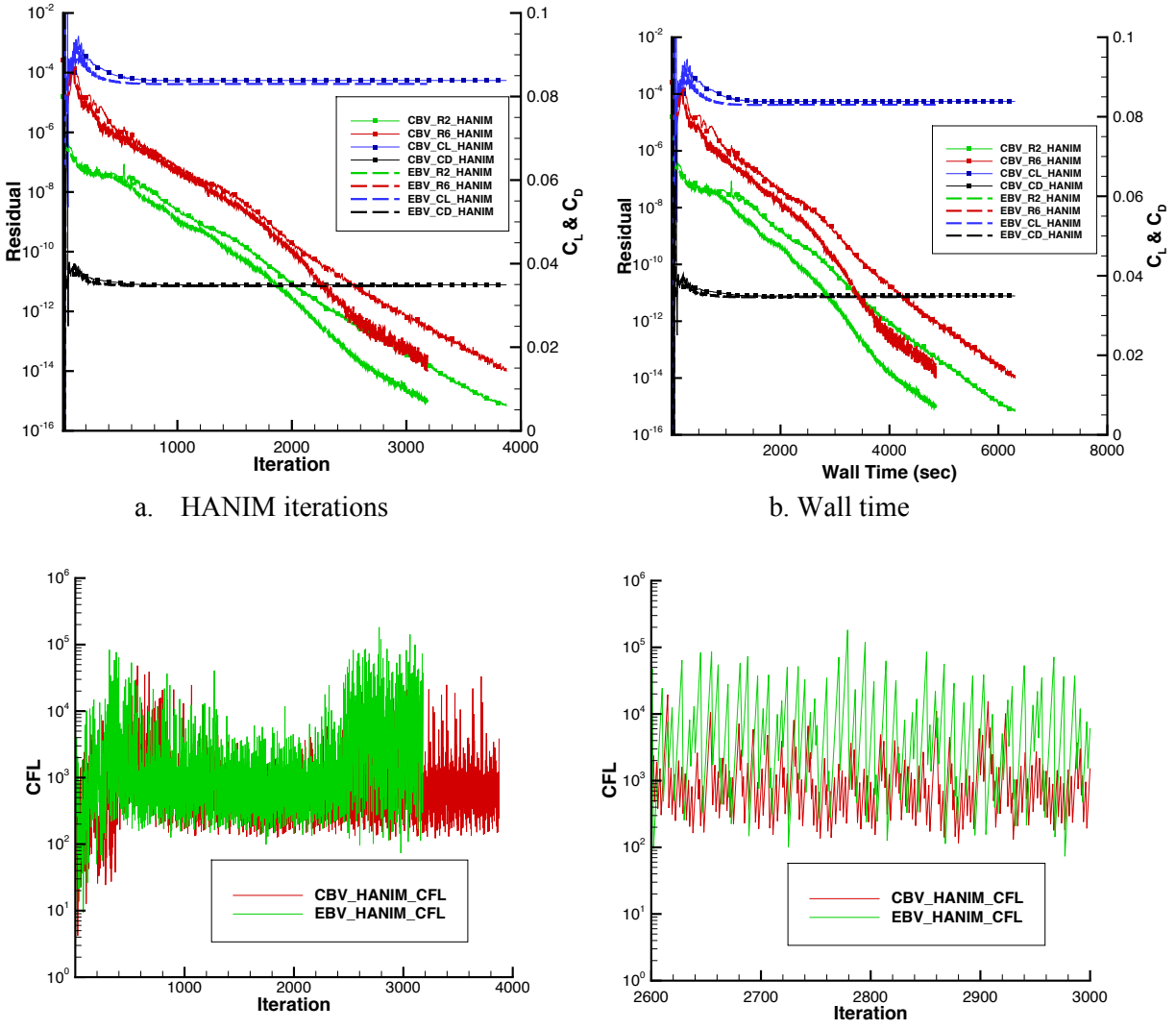
Grid	Meanflow viscous fluxes and SA-neg diffusion: residuals only			Complete nonlinear iteration		
	CBV	EBV	EBV speedup	CBV	EBV	EBV speedup
T1	0.084	0.023	73%	0.687	0.631	8%
T2	0.081	0.022	73%	0.656	0.602	8%
T3	0.056	0.015	73%	0.373	0.373	10%

HANIM iterations

The strong nonlinear iterative solver, HANIM [6, 7], presents unique challenges and opportunities for speedup of the EBV method. The desire to run at as high a CFL number as possible leads to an increased number of preconditioner sweeps in the linear solver, possibly multiple search directions for the GCR solver, and relatively frequent nonlinear iteration failures, which tend to decrease the fraction of computations related to viscous/diffusion terms, thus limiting opportunities for the EBV speedup. However, each HANIM iteration performs Jacobian evaluation, which increases opportunities for the EBV speedup. The precise prediction of the EBV speedup for an individual HANIM iteration is not possible because HANIM iterations are not identical. There are many run-time decisions that HANIM makes based on comparison of floating-point numbers. Thus, the nonlinear convergence in each iteration is sensitive to small details of discretization, previous solution, and solver parameters. The CBV and EBV residual convergence plots are expected to be visibly different, unlike the baseline iteration plots shown in Fig. 3a.

HANIM simulations have been conducted for both the CBV and EBV methods on the T2 grid. The EBV solutions use 20 EBV coefficients per edge. The following HANIM parameters are used. The maximum number of preconditioner sweeps is set to 300 for both meanflow and turbulence. The target preconditioner residual reduction is set to 0.2. The GCR residual reduction target is set to 0.92. Only one GCR search direction is allowed.

Figure 4 illustrates convergence of the HANIM-EBV and HANIM-CBV iterations. Figures 4a and 4b show the convergence history of residuals and aerodynamic coefficients versus HANIM iterations and versus wall time, respectively. The thin lines with symbols show convergence of the HANIM-CBV iterations and the dashed thicker lines show convergence of the HANIM-EBV iterations. The aerodynamic coefficients computed by HANIM iterations converge to the same values reported in Table 2. Figures 4c and 4d show global and zoomed views of the CFL history. As expected, the HANIM-CBV and HANIM-EBV convergence histories versus iterations shown in Fig. 4a are visibly different. Both HANIM-EBV and HANIM-CBV iterations converge well, achieve an adaptive CFL number that is significantly higher than the CFL number of 100 prescribed for the baseline iterations, and dramatically reduce the time to converge residuals in comparison to the baseline iterations (cf. Fig. 3). In this test, the HANIM-EBV solver runs at a somewhat higher CFL number than the HANIM-CBV solver. As a result, HANIM-EBV converges the rms norm of residuals to the stopping tolerance of 10^{-14} in less than 3200 iterations, 20% fewer iterations than those used by HANIM-CBV and requires significantly less time to converge residuals and aerodynamic coefficients.



a. HANIM iterations

b. Wall time

c. CFL number (global view)

d. CFL number (zoomed view)

Figure 4. HANIM iterations on tetrahedral grids.

The statistics of the baseline and HANIM iterations on the T2 grid is shown in Table 8. The HANIM convergence time and iteration count are compared with the time and iteration count of the baseline solver documented in Table 3. HANIM dramatically speeds up the solution process. In this test, HANIM-EBV converged 23% faster than HANIM-CBV. An average HANIM-EBV iteration takes only 7% less time than an average HANIM-CBV iteration. This modest speedup per iteration is expected as the fraction of viscous computations in a HANIM iteration is reduced comparing to a baseline iteration. The significant overall speedup demonstrated by the HANIM-EBV solver is attributed to the effect of higher CFL number on convergence of nonlinear iterations. In general, there is no expectation that HANIM-EBV iterations can universally run at higher CFL number and be faster than HANIM-CBV iterations. HANIM iterations with any viscous-flux method are expected to be superior to the corresponding baseline iterations.

Table 8. Nonlinear iterations on T2 grid.

	Baseline iterations		HANIM iterations		HANIM speedup
	Iterations	**Wall time (sec)**	**Iterations**	**Wall time (sec.)**	
CBV	42528	30174	3875	6304	4.79
EBV	42308	26502	3182	4844	5.47
EBV speedup		12%		23%	

Further performance improvements on tetrahedral grids

While the EBV performance shows a significant improvement over the CBV performance, further cost reduction is possible for both methods. Tables 4 and 5 show a much higher EBV speedup for SA-neg computations than for meanflow computations. This mismatch is explained by a suboptimal CBV implementation of the SA-neg residual and Jacobian. The CBV evaluation of the SA-neg diffusion term takes more than three times longer in comparison with the CBV evaluation of the meanflow viscous fluxes. The FUN3D-FV CBV method for the meanflow equations is highly optimized for tetrahedral cells, while the SA-neg CBV implementation does not take advantage of such optimization. In particular, the CBV evaluation on a general cell includes gradient augmentation at each edge of the cell. Such augmentation is expensive, but it is required to ensure stability of the discretization scheme on nontetrahedral cells. The augmentation is not needed for tetrahedra and can be skipped resulting in much less expensive gradient evaluation. The meanflow CBV implementation segregates tetrahedra from other cells and performs gradient evaluation optimally. The SA-neg CBV implementation is agnostic to the cell type, performing gradient augmentation for all cells.

Optimization of the EBV memory requirements is possible. Instead of 20 EBV coefficients per edge, only seven coefficients should be stored for interior edges. This optimization should allow better use of memory and further speedup EBV solutions. Reduction of the memory footprint makes the EBV scheme more suitable for modern peta- and exascale computer architectures, where the memory bandwidth remains the main bottleneck for memory bound computations, such as FUN3D solutions.

From Tables 4 and 5, the cost of meanflow Jacobian evaluation for the CBV and EBV methods exceeds the cost of the corresponding residual evaluation by a factor of six. In general, the cost of Jacobian evaluation is expected to be comparable with the cost of residual evaluation. The FUN3D-FV Jacobian is computed with respect to the conservative variables, $(\rho, \rho\boldsymbol{u}, E)$, while the meanflow residuals are evaluated from the primitive variables, $(\rho, \boldsymbol{u}, p)$. Part of the cost increase can be explained by additional conversions between the primitive and conservative variables that are embedded into Jacobian evaluation. The EBV method for the SA-neg equation does not need such conversions and exhibits comparable timing for the diffusion contributions to the Jacobian and the residual. However, the six-fold cost increase for the meanflow Jacobian over the corresponding residual seems excessive, even if the conversions between the primitive and the conservative variables are taken into account. Significant cost reduction should be possible through optimization of the Jacobian implementation.

Table 4 shows that the cost of the EBV SA-neg diffusion is about 30% less than the cost of the EBV meanflow viscous fluxes. Intuitively, the ratio is expected to be higher. The SA-neg model is one equation, while four meanflow equations have viscous fluxes. It is possible that additional performance improvement can be achieved by further optimizing the EBV SA-neg diffusion implementation.

The EBV method that is used in this paper is implemented in a separate edge-based loop. This approach has been temporarily adopted because it simplifies integration of the EBV method with multiple inviscid-flux and turbulence-model options available in FUN3D. This separate-loop implementation degrades the benefits of the EBV method. An exploratory single-processor simulation has been conducted on the T2 grid with the meanflow inviscid and EBV fluxes computed in a single edge-based loop. The meanflow Jacobian computations have also been performed in a single loop. In this simulation, Intel® FORTRAN 2019 compiler uses "-O3 -inline all" compiler options. The EBV speedup for the viscous-flux contributions to the meanflow residuals and Jacobian has increased from 35%-42% reported in Tables 4 and 5 to over 60%.

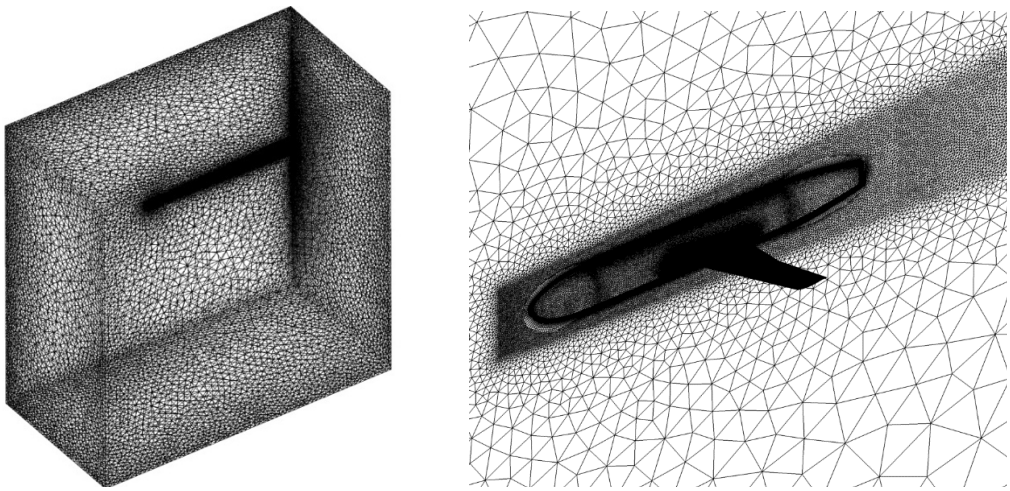
VI. EBV Solutions on Mixed-Element Grids

In this section, hybrid EBV/CBV solutions are compared with the CBV solutions on a mixed-element grid for a flow around a NASA juncture-flow model (JFM) configuration. This flow has been extensively studied experimentally and computationally [7, 41-44] and has been chosen as a verification case for the upcoming High-Fidelity CFD Workshop 2022[‡] [45]. It has been convincingly established that linear turbulence models, such as the SA-neg model, cannot capture important characteristics of flow separation typical for juncture flows. The goal of the solutions reported here is not to analyze the actual flow phenomena, but rather to demonstrate feasibility of hybrid EBV/CBV solutions on practical mixed-element grids and to establish similarity between the hybrid EBV/CBV solutions and well-verified baseline CBV solutions.

The flow conditions for this test are the following: the freestream Mach number is 0.189, the Reynolds number is 2,400,000 based on the crank chord of 557.17mm, the reference temperature is 288.84 Kelvin, and the angle of attack is 5 degrees. A family of grids have been generated for the JFM configuration by Pointwise® using the Glyph script package GeomToMesh [46]. The specific mixed-element grid chosen for the study is Grid I [7, 44] that has 13,036,210 grid points, 16,645,072 tetrahedra, 20,368,758 prisms, and 112,989 pyramids. The volume and surface mesh are shown in Fig. 5.

Recall that in hybrid EBV/CBV computations on mixed-element grids, the meanflow viscous fluxes and SA-neg diffusion term are evaluated by the EBV method on tetrahedra and by CBV method on prisms and pyramids. The EBV computations are conducted in an edge-based loop over edges of tetrahedral cells, and the CBV computations are conducted in a cell-based loop over prisms and pyramids.

For the baseline iterations, the meanflow CFL of 50 and the SA-neg CFL of 30 are set to ensure convergence of all residuals to the level of 10^{-10}. The preconditioner performs 30 multicolor GS sweeps for the meanflow and SA-neg linear equations. For HANIM iterations, the following parameters are set. The maximum number of GS sweeps is 100 for the meanflow and SA-neg preconditioner equations. The preconditioner residual reduction target is 0.5, and the GCR residual reduction target is 0.92. Only one GCR search direction is allowed. The mixed-element solutions have been computed using 60 Intel® Xeon Sandy Bridge compute nodes (960 cores). Figures 6 and 7 illustrate convergence of the baseline and HANIM iterations, respectively.



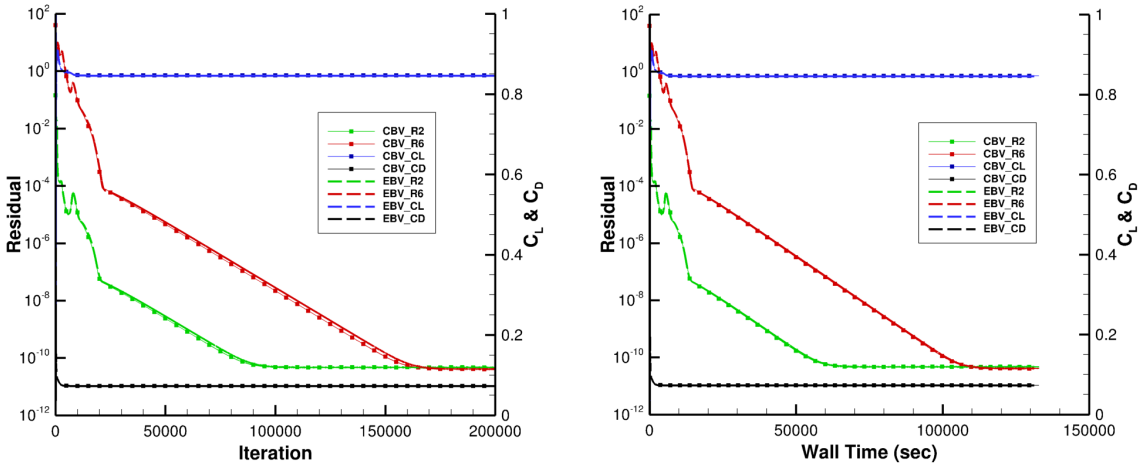
a. Volume mesh b. Surface mesh

Figure 5. Volume and surface mesh for Grid I of juncture flow model.

Since a point-based graph partitioner is generally agnostic to cell type, load imbalances related to cell-specific operations are often encountered in practice. For this test case, the 960 partitions are well balanced

[‡] https://turbmodels.larc.nasa.gov/highfidelitycfd_workshop2022.html; accessed May 18, 2021

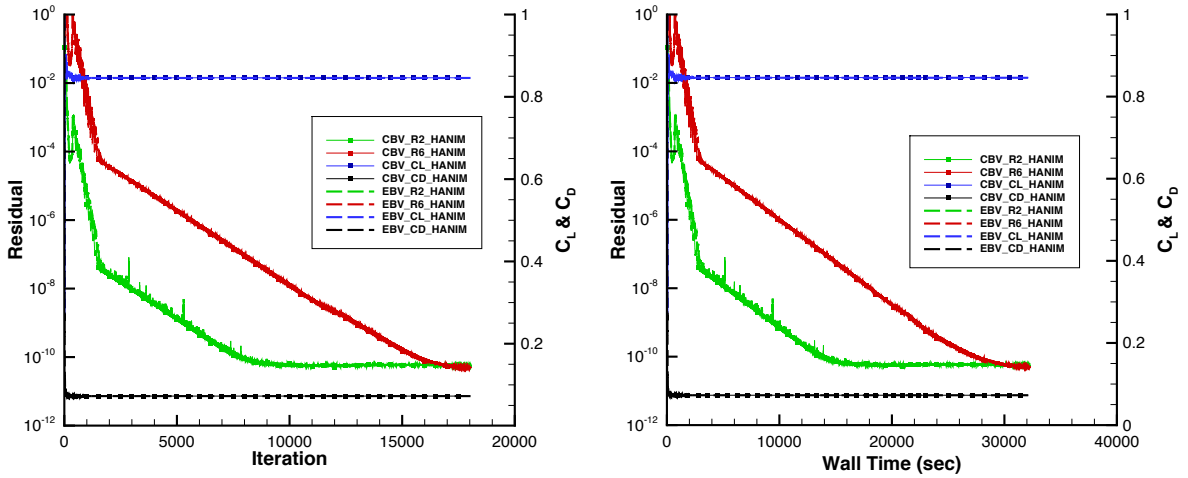
for the grid points; the ratio of maximum to minimum grid points per partition is 1.34. However, there are three partitions that have only prismatic cells. For this reason, there is no expectation of improved efficiency for the parallel EBV/CBV solutions in which the EBV method is applied only on tetrahedra.



a. Nonlinear iterations b. Wall time

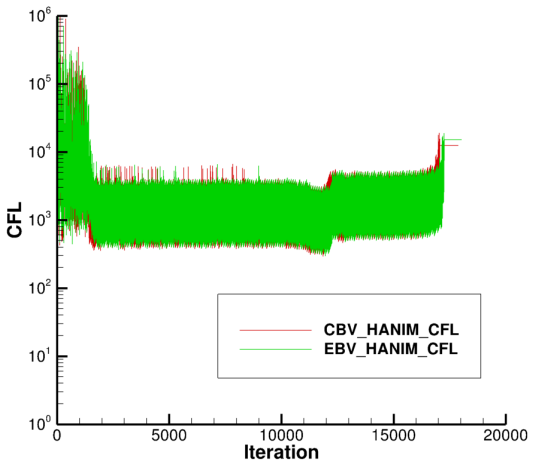
Figure 6. Baseline iterations on JFM mixed-element grid.

As expected, convergence of residuals and lift and drag coefficients observed in the baseline EBV and CBV iterations is almost identical both in terms of iterations and wall time. HANIM solvers converge to the same solutions much faster. Both HANIM solvers dramatically outperform the baseline solvers, using an order of magnitude fewer iterations and reducing the time to convergence by more than 70%. The difference between the HANIM-EBV and HANIM-CBV convergence plots is small; the iteration and time plots shown in Figs. 7a and 7b, respectively, are hardly distinguishable in the global view. Figures 7c and 7d indicate that HANIM iterations operate with an average CFL number above 1000, which is more than an order of magnitude higher than the operational CFL numbers set for the baseline iterations.

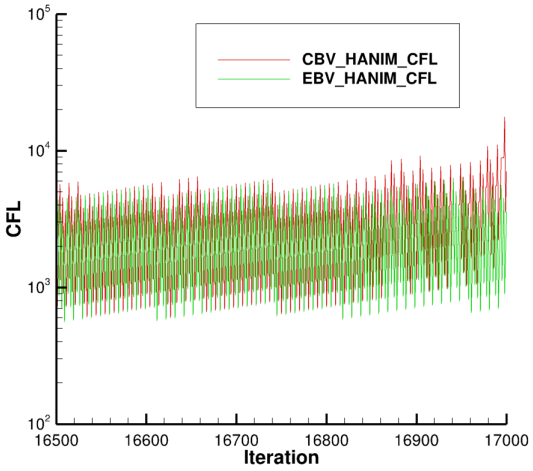


a. HANIM iterations b. Wall time

Figure 7. HANIM iterations on JFM mixed-element grid.



c. CFL number (global view)



d. CFL number (zoomed view)

Figure 7. Concluded.

VII. Concluding Remarks

An efficient, conservative, edge-based viscous (EBV) method for evaluation of meanflow viscous fluxes and turbulence-model diffusion terms in the Reynolds-averaged Navier-Stokes (RANS) equations with the negative variant of the one-equation Spalart-Allmaras (SA-neg) turbulence model has been developed. The EBV method has been derived for tetrahedral cells and implemented in FUN3D, NASA's node-centered, unstructured-grid RANS solver. The EBV method reformulates the extensively verified and validated baseline second-order finite-volume (FUN3D-FV) method for viscous fluxes. The baseline formulation, which is equivalent to the finite-element Galerkin formulation on tetrahedral grids, uses cell-based Green-Gauss gradients, evaluates viscous fluxes in a cell-based loop, and is referred to as the cell-based viscous (CBV) method. The EBV method recombines operations to compute Green-Gauss gradients by edge, which leads to an equivalent formulation that can be computed in an edge-based loop. The resulting EBV implementation eliminates redundancies inherent in the CBV method, improves efficiency, and preserves favorable properties of the CBV method, such as a compact stencil based on nearest neighbors. A hybrid EBV/CBV method has been implemented for simulations on mixed-element grids; the EBV method is performed for tetrahedra and the CBV method is performed for cells of other types.

The EBV method has been verified and assessed using a benchmark subsonic separated flow around a hemisphere-cylinder configuration. The NASA turbulence model resource (TMR) website provides flow conditions, families of grids, and reference solutions for this case. The EBV and CBV solutions have been compared on a family of tetrahedral grids. The residual reduction per iteration and final solutions computed with the EBV and CBV methods are almost identical. The EBV method reduces the time for evaluation of the meanflow viscous fluxes and corresponding Jacobian by 35%-42%. The FUN3D-FV implementation of the CBV method for meanflow fluxes and Jacobian has previously been optimized, which makes this significant EBV efficiency gain more impressive. For evaluation of the diffusion term of the SA-neg turbulence model and the corresponding Jacobian contributions, the EBV speedup is much higher, 85%-93%. For the simple baseline nonlinear iterations, this EBV speedup is translated into 8%-23% reduction in the time to solution, depending on how often nonlinear iterations perform Jacobian updates. A strong hierarchical adaptive nonlinear iteration method (HANIM) has recently been implemented to improve robustness of FUN3D-FV solutions and accelerate convergence. HANIM updates the Jacobian at each nonlinear iteration, which increases the fraction of time spent on viscous-flux computations and increases the opportunity for EBV gains. On the other hand, HANIM also performs additional computations unrelated to the meanflow viscous fluxes and the SA-neg diffusion term. For the tests considered in this paper, the EBV method proved to be beneficial for HANIM convergence, reducing the average iteration cost and the

number of iterations to convergence. An additional 23% reduction in time to solution has been observed with HANIM-EBV in comparison to a highly efficient HANIM-CBV solution.

Opportunities for further efficiency improvements for both the EBV and CBV methods on tetrahedral grids have been identified. Significant reduction of the EBV memory footprint, from twenty to seven EBV coefficients per edge, should improve memory access and further reduce the computation time. Optimization of the SA-neg diffusion implementation can significantly speed up the baseline CBV method. Optimization of the meanflow Jacobian evaluation and the SA-neg diffusion term can accelerate both the CBV and EBV solutions. Currently, the EBV method has been implemented in a separate edge-based loop to simplify logistics of integration with the rest of FUN3D. Preliminary tests indicate that combining viscous and inviscid fluxes in a single edge-based loop can improve the EBV performance by an additional 20%.

A hybrid EBV/CBV method has been applied to compute solutions on a practical mixed-element grid generated around a NASA juncture-flow model configuration. The EBV method has been applied on tetrahedra and the CBV method has been applied on cells of other types. The hybrid EBV/CBV solutions proved to be accurate, matched the CBV solutions and convergence history almost perfectly. However, the hybrid method produced no efficiency gains since the domain decomposition methods used in the study are solely based on equidistribution of point-based operations. Development of an EBV method for cells of all types is expected to address this deficiency and extend the EBV efficiency gains to solutions on mixed-element grids.

Appendix: Conservation Property of CBV and EBV Methods

A property of conservation for a discretization method can be understood through relations between discrete residuals defined on a set of nonoverlapping control volumes and conserved solution quantities. A conservative residual at any control volume can be represented as a linear combination of conserved solution quantities. A solution quantity is conserved if any perturbation of this quantity in the interior of the computational domain leads to the net zero changes in the sum of all residuals. A finite-volume discretization method is an example of a conservative discretization method. A finite-volume residual at a control volume is a summation of fluxes (conserved quantities) through the control-volume boundary. Any perturbation of a flux at the boundary that separates two control volumes changes residuals at both control volumes; the changes are equal in magnitude and opposite in sign.

In this appendix, we illustrate the conservation property of the CBV and EBV discretization methods. A scalar three-dimensional linear diffusion operator of a twice differentiable function $\varphi(x, y)$ is defined as

$$\boldsymbol{\nabla}(\mu \boldsymbol{\nabla} \varphi) \tag{38}$$

where μ is a spatially variable diffusion coefficient. The operator is discretized on a general tetrahedron with points $\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{p}_3,$ and $\boldsymbol{p}_4$ shown in Fig. 8. The point $\boldsymbol{m}_{ij} = \frac{\boldsymbol{p}_i + \boldsymbol{p}_j}{2}$ is the median of the edge connecting points $\boldsymbol{p}_i$ and $\boldsymbol{p}_j$. The point $\boldsymbol{f}_{ijk} = \frac{\boldsymbol{p}_i + \boldsymbol{p}_j + \boldsymbol{p}_k}{3}$ is the centroid of the face that has points $\boldsymbol{p}_i, \boldsymbol{p}_j,$ and $\boldsymbol{p}_k$. The point $\boldsymbol{c}_{1234} = \frac{\boldsymbol{p}_1 + \boldsymbol{p}_2 + \boldsymbol{p}_3 + \boldsymbol{p}_4}{4}$ is the centroid of the tetrahedron.

The quadrilateral shape shaded in Fig. 8a is the portion of the control-volume boundary that is shared by the control volumes centered at the points $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$; $\boldsymbol{d}_{12}$ is the corresponding directed-area vector pointing outward from the control volume centered at the point $\boldsymbol{p}_1$. In the CBV method, the flux computed at the control-volume boundary shared between the points $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ provides the following contribution to the residual at the point $\boldsymbol{p}_1$

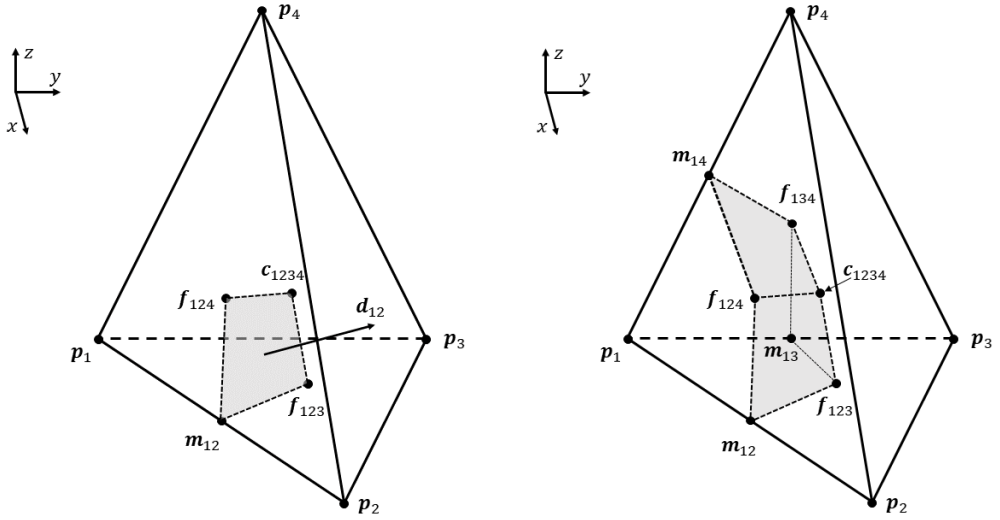
$$R_1 = R_1 + \mu_{1234}(\boldsymbol{\nabla}_{1234}\varphi \cdot \boldsymbol{d}_{12}). \tag{39}$$

Here, the diffusion coefficient and the gradient are evaluated at the tetrahedron from function values defined at points as

$$\mu_{1234} = \frac{\mu_1 + \mu_2 + \mu_3 + \mu_4}{4}; \tag{40}$$

$$\boldsymbol{\nabla}_{1234}\varphi = \frac{1}{Vol}\left[\frac{\varphi_2 + \varphi_3 + \varphi_4}{3}\boldsymbol{n}_1 + \frac{\varphi_1 + \varphi_3 + \varphi_4}{3}\boldsymbol{n}_2 + \frac{\varphi_1 + \varphi_2 + \varphi_4}{3}\boldsymbol{n}_3 + \frac{\varphi_1 + \varphi_2 + \varphi_3}{3}\boldsymbol{n}_4\right]. \tag{41}$$

This gradient discretization is the same as in Eq. 18; Vol is the volume of the tetrahedron, the vector $\boldsymbol{n}_i$ is the outward directed area of the triangular face opposite to the point $\boldsymbol{p}_i$.



a. CBV method b. EBV method

Figure 8. Control-volume boundaries within tetrahedron.

The contribution to the residual at the point $\boldsymbol{p}_2$ from the flux computed at the shared portion of the control-volume boundary has the same magnitude and opposite sign as the outward directed area is pointing to the opposite direction.

$$R_2 = R_2 - \mu_{1234}(\boldsymbol{\nabla}_{1234}\varphi \cdot \boldsymbol{d}_{12}). \tag{42}$$

Thus, the CBV discretization is conservative.

The EBV method conserves edge contributions, $\mu_{ij}\Delta_{ij}\varphi$, for which the edge-based diffusion coefficient is defined similarly to Eq. 30, and the edge-based solution difference is defined similarly to Eq. 20:

$$\mu_{ij} = \frac{\mu_i + \mu_j}{2}, \qquad \Delta_{ij}\varphi = \varphi_i - \varphi_j. \tag{43}$$

Since the edge-based solution difference $\Delta_{ij}\varphi$ contributes to the cell gradient, to compute the full contribution from the edge-based term to the residual at the point $\boldsymbol{p}_i$, one must integrate over the entire control-volume boundary located within the cell. The shaded area in Fig. 8b indicates the portion of the control-volume boundary within the tetrahedron that corresponds to the point $\boldsymbol{p}_1$. Recall that the combined directed area of the shaded control-volume boundary relates to the area of the opposite face as in Eq. 26:

$$\boldsymbol{d}_1 = \frac{1}{3}\boldsymbol{n}_1. \tag{44}$$

For contributions to the residual at the point $\boldsymbol{p}_1$, the gradient is represented according to Eq. 21a as

$$\boldsymbol{\nabla}_{1234}\varphi = \frac{-1}{3\,Vol}(\Delta_{21}\varphi\boldsymbol{n}_2 + \Delta_{31}\varphi\boldsymbol{n}_3 + \Delta_{41}\varphi\boldsymbol{n}_4). \tag{45}$$

The EBV method provides the following contribution to the residual at the point $\boldsymbol{p}_1$ from the tetrahedron:

$$\begin{aligned} R_1 &= R_1 - \frac{1}{3\,Vol}\left(\mu_{21}\Delta_{21}\varphi(\boldsymbol{n}_2 \cdot \boldsymbol{d}_1) + \mu_{31}\Delta_{31}\varphi(\boldsymbol{n}_3 \cdot \boldsymbol{d}_1) + \mu_{41}\Delta_{41}\varphi(\boldsymbol{n}_4 \cdot \boldsymbol{d}_1)\right) \\ &= R_1 - \frac{1}{9\,Vol}\left(\mu_{21}\Delta_{21}\varphi(\boldsymbol{n}_2 \cdot \boldsymbol{n}_1) + \mu_{31}\Delta_{31}\varphi(\boldsymbol{n}_3 \cdot \boldsymbol{n}_1) + \mu_{41}\Delta_{41}\varphi(\boldsymbol{n}_4 \cdot \boldsymbol{n}_1)\right). \end{aligned} \tag{46}$$

Specifically, the contribution to the residual at the point $\boldsymbol{p}_1$ from the edge connecting the points $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ is $\frac{\mu_{21}\Delta_{21}\varphi}{9\,Vol}(\boldsymbol{n}_2 \cdot \boldsymbol{n}_1)$. Analogously, the contribution to the residual at the point $\boldsymbol{p}_2$ from this edge is $\frac{\mu_{12}\Delta_{12}\varphi}{9\,Vol}(\boldsymbol{n}_2 \cdot \boldsymbol{n}_1)$. From Eq. 43, $\mu_{21} = \mu_{12}$ and $\Delta_{21}\varphi = -\Delta_{12}\varphi$, implying that the contributions to the residuals at the points $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ from the edge connecting these points are equal in magnitude and opposite in sign. Thus, the EBV discretization of the scalar operator (Eq. 38) is conservative.

The strict conservation property for a discretized system of conservation-law equations requires conservation (i.e., preservation of the sum of all residuals over the computational domain) for each individual equation. To demonstrate the conservation property for the system of momentum equations, one can consider an interior edge of a tetrahedral grid. The edge is surrounded by tetrahedra. The union of the tetrahedra forms a closed polyhedron. The number of faces in such a polyhedron is twice the number of the tetrahedra surrounding the edge. A simple example of three tetrahedra that surround and share an edge is shown in Figure 9.

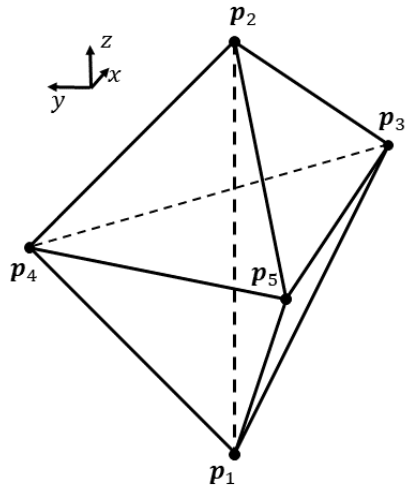


Figure 9. Three tetrahedra share edge $[\boldsymbol{p}_1, \boldsymbol{p}_2]$.

In a general construction, where n tetrahedra surround and share an edge, one can assume that the shared edge is $[\boldsymbol{p}_1, \boldsymbol{p}_2]$. All other grid points of the involved tetrahedra are ordered from $\boldsymbol{p}_3$ to $\boldsymbol{p}_{n+2}$. The point $\boldsymbol{p}_i$ is connected by edge to the points $\boldsymbol{p}_{i-1}$ and $\boldsymbol{p}_{i+1}$, $i = 4, \dots, n+1$; and the points $\boldsymbol{p}_3$ and $\boldsymbol{p}_{n+2}$ are connected by edge to form a closed polyhedron. By construction, the index i uniquely identifies tetrahedron formed by points $\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{p}_i$, and $\boldsymbol{p}_{i+1}, i = 3, \dots, n+1$ (or by points $\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{p}_{n+2}$, and $\boldsymbol{p}_3$, if $i = n+2$). For tetrahedron i, the directed area vector opposite to the point $\boldsymbol{p}_1$ is denoted as $\boldsymbol{n}_{i1}$

$$\boldsymbol{n}_{i1} = \frac{1}{2}[\boldsymbol{p}_i - \boldsymbol{p}_2] \times [\boldsymbol{p}_{i+1} - \boldsymbol{p}_2];$$

the directed area vector opposite to the point $\boldsymbol{p}_2$ is denoted as $\boldsymbol{n}_{i2}$

$$\boldsymbol{n}_{i2} = \frac{1}{2}[\boldsymbol{p}_{i+1} - \boldsymbol{p}_1] \times [\boldsymbol{p}_i - \boldsymbol{p}_1];$$

and the volume of the tetrahedron i is denoted as

$$Vol_i = \frac{1}{3}(\boldsymbol{n}_{i1} \cdot [\boldsymbol{p}_2 - \boldsymbol{p}_1]) = \frac{1}{3}(\boldsymbol{n}_{i2} \cdot [\boldsymbol{p}_1 - \boldsymbol{p}_2]);$$

The collective contribution to the momentum-conservation residual at the grid point $\boldsymbol{p}_1$ (cf. Eq. 31) from all tetrahedra that share the edge $[\boldsymbol{p}_1, \boldsymbol{p}_2]$ can be expressed as

$$\boldsymbol{R}_{m1} = \boldsymbol{R}_{m1} + \frac{(\mu + \mu_t)_{21}}{9V_1}\Delta_{21}\boldsymbol{u}\left(\sum_i \frac{1}{Vol_i}\left[(\boldsymbol{n}_{i2} \cdot \boldsymbol{n}_{i1})\boldsymbol{I} - \frac{2}{3}\boldsymbol{n}_{i1}\boldsymbol{n}_{i2}^{\mathrm{T}} + \boldsymbol{n}_{i2}\boldsymbol{n}_{i1}^{\mathrm{T}}\right]\right). \quad (47)$$

The summation is over all tetrahedra that share the edge $[\boldsymbol{p}_1, \boldsymbol{p}_2]$. The matrix of collective EBV coefficients corresponding to the edge is computed as

$$\boldsymbol{C}_{2,1} = \sum_i \frac{1}{Vol_i}\left[(\boldsymbol{n}_{i2} \cdot \boldsymbol{n}_{i1})\boldsymbol{I} - \frac{2}{3}\boldsymbol{n}_{i1}\boldsymbol{n}_{i2}^{\mathrm{T}} + \boldsymbol{n}_{i2}\boldsymbol{n}_{i1}^{\mathrm{T}}\right]. \quad (48)$$

To satisfy the strict conservation property, the matrix of collective EBV coefficients should be symmetric.

As mentioned in Section IV, the matrix contributions from individual tetrahedra are not symmetric. For the matrix of collective EBV coefficients to be symmetric it is necessary and sufficient to show that the matrix

$$\boldsymbol{M} = \sum_i \frac{\boldsymbol{n}_{i1}\boldsymbol{n}_{i2}^{\mathrm{T}}}{Vol_i} \tag{49}$$

is symmetric for an arbitrary set of tetrahedra that surround and share the edge $[\boldsymbol{p}_1, \boldsymbol{p}_2]$. While a rigorous proof of this statement is not available yet, examples below indicate that the property is expected to be satisfied on general tetrahedral grids.

For example, for the grid system shown in Figure 9, Tables 9, 10, and 11 specify the coordinates of the points, the directed area vectors, and the inverse volumes of tetrahedra, respectively.

Table 9. Point coordinates for grid system in Fig. 9.

Point	x	y	z
$\boldsymbol{p}_1$	0	0	0
$\boldsymbol{p}_2$	0	0	2
$\boldsymbol{p}_3$	1	0	1.5
$\boldsymbol{p}_4$	0	1	1
$\boldsymbol{p}_5$	-0.5	-0.5	1

Table 10. Directed area vectors for grid system in Fig. 9.

Point	x	y	z
$\boldsymbol{n}_{31}$	0.25	0.5	0.5
$\boldsymbol{n}_{41}$	-0.75	0.25	0.25
$\boldsymbol{n}_{51}$	0.125	-0.625	0.25
$\boldsymbol{n}_{32}$	0.75	0.5	-0.5
$\boldsymbol{n}_{42}$	-0.75	0.25	-0.25
$\boldsymbol{n}_{52}$	0.375	-0.875	-0.25

Table 11. Inverse volumes of tetrahedra for grid system in Fig. 9.

i	Tetrahedra	$1/Vol_i$
3	$\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{p}_3, \boldsymbol{p}_4$	3
4	$\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{p}_4, \boldsymbol{p}_5$	6
5	$\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{p}_5, \boldsymbol{p}_3$	6

The corresponding tensor products are

$$\begin{aligned}
\boldsymbol{n}_{31}\boldsymbol{n}_{32}^{\mathrm{T}} &= \begin{bmatrix} 0.1875 & 0.125 & -0.125 \\ 0.375 & 0.25 & -0.25 \\ 0.375 & 0.25 & -0.25 \end{bmatrix}; \\
\boldsymbol{n}_{41}\boldsymbol{n}_{42}^{\mathrm{T}} &= \begin{bmatrix} 0.5625 & -0.1875 & -0.125 \\ -0.1875 & 0.0625 & -0.0625 \\ -0.1875 & 0.0625 & -0.0625 \end{bmatrix}; \\
\boldsymbol{n}_{51}\boldsymbol{n}_{52}^{\mathrm{T}} &= \begin{bmatrix} 0.046875 & -0.109375 & -0.03125 \\ -0.234375 & 0.546875 & 0.15625 \\ 0.09375 & -0.21875 & -0.0625 \end{bmatrix}.
\end{aligned} \tag{50}$$

For this system, the matrix $\boldsymbol{M}$ is evaluated as

$$\boldsymbol{M} = \frac{\boldsymbol{n}_{31}\boldsymbol{n}_{32}^{\mathrm{T}}}{Vol_3} + \frac{\boldsymbol{n}_{41}\boldsymbol{n}_{42}^{\mathrm{T}}}{Vol_4} + \frac{\boldsymbol{n}_{51}\boldsymbol{n}_{52}^{\mathrm{T}}}{Vol_5} = \begin{bmatrix} 4.21875 & -1.40625 & 0.5625 \\ -1.40625 & 4.40625 & -0.1875 \\ 0.5625 & -0.1875 & -1.5 \end{bmatrix}. \quad (51)$$

In a more practical example, the matrix of EBV coefficients is evaluated on a tetrahedral grid for the NASA high-lift common research model used for the AIAA CFD High Lift Prediction Workshop[§]. A randomly chosen interior edge $[\boldsymbol{p}_{30}, \boldsymbol{p}_{66}]$ is surrounded by six tetrahedra. The eight grid points forming this local subgrid are listed in Table 12. The six tetrahedra that share the edge $[\boldsymbol{p}_{30}, \boldsymbol{p}_{66}]$ are listed in Table 13.

Table 12. Point coordinates for practical sub-grid system.

Grid point number	x	y	z
28	1769.57819	100.27637	290.63749
29	1775.77782	100.27391	290.63578
30	1769.58102	96.62027	295.63767
64	1775.78144	96.62027	295.63767
65	1769.57819	100.27944	290.63961
66	1775.77782	100.27637	290.63748
69	1769.58102	96.62323	295.63995
71	1775.78144	96.62323	295.63995

Table 13. Tetrahedra that share edge $[\boldsymbol{p}_{30}, \boldsymbol{p}_{66}]$.

Tetrahedra	Point 1	Point 2	Point 3	Point 4
243	28	30	29	66
253	30	66	28	65
261	30	64	29	66
268	66	30	64	71
272	71	66	30	69
273	66	65	30	69

The matrix of collective EBV coefficients associated with the edge $[\boldsymbol{p}_{30}, \boldsymbol{p}_{66}]$ is symmetric:

$$\boldsymbol{C}_{30,66} = \begin{bmatrix} 0.00161201 & 0.278009 & 0.202816 \\ 0.278009 & 0.329915 & -0.104578 \\ 0.202816 & -0.104578 & 0.00140457 \end{bmatrix}. \quad (52)$$

The required EBV storage for this edge is the total of seven coefficients: six coefficients are needed for the momentum-conservation residuals and one coefficient is needed for the heat flux; this latter coefficient is also suitable for the EBV discretization of the SA-neg diffusion.

The matrix of collective EBV coefficients is symmetric only if the edge is fully interior, i.e., cells that share the edge form a closed shape around the edge, and all cells sharing the edge are tetrahedra. For boundary edges and edges that are shared by cells of different types, nine EBV coefficients are needed to compute the momentum-conservation residuals resulting in the total of ten EBV coefficients per edge.

Acknowledgments

The Transformative Tools and Technologies (TTT) project of the Transformative Aeronautics Concepts Program and the NASA Revolutionary Vertical Lift Technology (RVLT) project within the NASA Aeronautics Research Mission Directorate partially funded the work reported here.

[§] https://hiliftpw.larc.nasa.gov; accessed May 18, 2021

References

[1] Biedron R. T., Carlson J. R., Derlaga J. M., Gnoffo P. A., Hammond D. P., Jones W. T., Kleb B., Lee-Rausch E. M., Nielsen E. J., Park M. A., Rumsey C. L., Thomas J. L., Thompson K. B., Walden, A. C., Wang, L., and Wood W. A., "FUN3D Manual: 13.7," NASA TM 2020-5010139, 2020.

[2] Anderson W. K. and Bonhaus D. L., "An Implicit Upwind Algorithm for Computing Turbulent Flows on Unstructured Grids," *Computers and Fluids*, Vol. 23, No. 1, 1994, pp. 1-21. https://doi.org/10.1016/0045-7930(94)90023-X

[3] Biedron R. T. and Thomas J. L., "Recent Enhancements to the FUN3D Flow Solver for Moving-Mesh Applications," AIAA 2009-1360. https://doi.org/10.2514/6.2009-1360

[4] Nielsen, E. J., and Diskin, B., "High-Performance Aerodynamic Computations for Aerospace Applications," *Parallel Computing*, Vol. 64, 2017, pp. 20–32. https://doi.org/10.1016/j.parco.2017.02.004

[5] Gnoffo P., "Updates to Multi-Dimensional Flux Reconstruction for Hypersonic Simulations on Tetrahedral Grids," AIAA 2010-1271. https://doi.org/10.2514/6.2010-1271.

[6] Pandya, M., Diskin, B., Thomas, J., and Frink, N., "Assessment of USM3D Hierarchical Adaptive Nonlinear Method Preconditioners for Three-Dimensional Cases," *AIAA Journal*, Vol. 55, No. 10, 2017, pp. 1–16. https://doi.org/10.2514/1.J055823.

[7] Wang L., Diskin B., Nielsen E. J., and Liu Y., "Improvements in Iterative Convergence of FUN3D Solutions," AIAA 2021-0857. https://doi.org/10.2514/6.2021-0857

[8] Wang L., Diskin B., Lopes L., Nielsen E. J., Lee-Rausch E., and Biedron R. T., "High-Fidelity Aero-Acoustic Optimization Tool for Flexible Rotors," *Journal of the American Helicopter Society,* Vol. 66, No. 2, 2021. https://doi.org/10.4050/JAHS.66.022004

[9] Kleb, B., Park, M. A., Wood, W. A., Bibb, K. L., Thompson, K. B., Gomez, R. J., III, and Tesch, S. H., "Sketch-to-Solution: An Exploration of Viscous CFD with Automatic Grids," AIAA 2019–2948. https://doi.org/10.2514/6.2019-2948.

[10] Walden A., Nielsen E. J., Diskin B., and Zubair M., "A Mixed Precision Multicolor Point-Implicit Solver for Unstructured Grids on GPUs," 2019 IEEE/ACM 9th Workshop on Irregular Applications: Architectures and Algorithms (IA3), Denver, CO, USA, 2019, pp. 23-30. https://doi.org/10.1109/IA349570.2019.00010

[11] Nastac G., Walden A., Nielsen E. J., and Frendi K., "Implicit Thermochemical Nonequilibrium Flow Simulations on Unstructured Grids using GPUs," AIAA 2021-0159. https://doi.org/10.2514/6.2021-0159

[12] Tannehill, J. C., Anderson, D. A., and Pletcher, R. H., "Computational Fluid Mechanics and Heat Transfer," third edition, Taylor & Francis, 2012.

[13] White, F. M., "Viscous Fluid Flow," McGraw Hill, New York, 1974, p. 28.

[14] Spalart, P. R. and Allmaras, S. R., "A One-Equation Turbulence Model for Aerodynamic Flows," *Recherche Aerospatiale*, No. 1, 1994, pp. 5-21.

[15] Allmaras, S. R., Johnson, F. T., and Spalart, P. R., "Modifications and Clarifications for the Implementation of the Spalart-Allmaras Turbulence Model," ICCFD7-1902, 7th International Conference on Computational Fluid Dynamics, Big Island, Hawaii, 9-13 July 2012.

[16] Roe, P. L., "Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes," *Journal of Computational Physics*, Vol. 43, No. 2, 1981, pp. 357–372. https://doi.org/10.1016/0021-9991(81)90128-5.

[17] Burg, C. O. E., "Higher Order Variable Extrapolation for Unstructured Finite Volume RANS Flow Solvers," AIAA Paper 2005–4999. https://doi.org/10.2514/6.2005-4999

[18] van Leer, B., "Towards the Ultimate Conservative Difference Scheme, V. A Second Order Sequel to Godunov's Method," *Journal of Computational Physics*, Vol. 32, No. 1, 1979, pp. 101–136. 10.1016/0021-9991(79)90145-1.

[19] Barth, T. J., "Numerical Aspects of Computing Viscous High Reynolds Number Flows on Unstructured Meshes," AIAA 91–0721. https://doi.org/10.2514/6.1991-721

[20] Haselbacher, A. C., *A Grid-Transparent Numerical Method for Compressible Viscous Flow on Mixed Unstructured Meshes*, Ph.D. thesis, Loughborough University, 1999.
[21] Nishikawa H., "Beyond Interface Gradient: A General Principle for Constructing Diffusion Schemes", AIAA 2010-5093. https://doi.org/10.2514/6.2010-5093
[22] Thomas J. L., Diskin B., and Nishikawa H., "A Critical Study of Agglomerated Multigrid Methods for Diffusion on Highly Stretched Grids," *Computers & Fluids,* Vol. 41, No. 1, 2011, pp. 82-93. https://doi.org/10.1016/j.compfluid.2010.09.023
[23] Carlson, J.-R., "Inflow/Outflow Boundary Conditions with Application to FUN3D," October 2011, NASA/TM–2011-217181, NASA Langley Research Center, Hampton, Virginia.
[24] van Leer, B. "Flux-Vector Splitting for the Euler Equations," ICASE Report 82-30, 1982.
[25] Pandya, M. J., Jespersen, D. C., Diskin, B., Thomas, J. L., and Frink, N. T., "Accuracy, Scalability, and Efficiency of Mixed-Element USM3D for Benchmark Three-Dimensional Flows," AIAA 2019–2333. https://doi.org/10.2514/6.2019-2333
[26] Pandya, M. J., Diskin, B., Thomas, J. L., and Frink, N. T., "Improved Convergence and Robustness of USM3D Solutions on Mix-Element Grids," *AIAA Journal*, Vol. 54, No. 9, 2016, pp. 2589–2596. https://doi.org/10.2514/1.J054545.
[27] Knoll, D. A. and Keyes, D. E., "Jacobian-Free Newton–Krylov Methods: A Survey of Approaches and Applications," *Journal of Computational Physics*, 193 (2004) 357–397. https://doi.org/10.1016/j.jcp.2003.08.010
[28] van der Vorst, H. A. and Vuik, C. "GMRESR: A Family of Nested GMRES Methods," *Numerical Linear Algebra with Applications*, Vol. 1, 1994, pp. 369-386. https://doi.org/10.1002/nla.1680010404
[29] Lucas, P., van Zuijlen, A. H., and Bijl, H., "Fast Unsteady Flow Computations with a Jacobian-free Newton-Krylov Algorithm," *Journal of Computational Physics*, Vol. 229, No. 24, 2010, pp. 9201-9215. https://doi.org/10.1016/j.jcp.2010.08.033
[30] Ceze, M. and Fidkowski, K., "A Robust Adaptive Solution Strategy for High-Order Implicit CFD Solvers," AIAA Paper 2011-3696. https://doi.org/10.2514/6.2011-3696
[31] Allmaras, S. R., Bussoletti, J. E., Hilmes, C. L., Johnson, F. T., Melvin, R. G., Tinoco, E. N., Venkatakrishnan, V., Wigton, L. B., and Young, D. P., "Algorithm Issues and Challenges Associated with the Development of Robust CFD Codes," In: *Variational Analysis and Aerospace Engineering, Springer Optimization and Its Applications*, Vol. 33, Springer, New York, NY, 2009, pp. 1-19. https://doi.org/10.1007/978-0-387-95857-6_1
[32] Katz, A. and Sankaran, V., "Mesh Quality Effects on the Accuracy of Euler and Navier-Stokes Solutions on Unstructured Meshes," *Journal of Computational Physics*, Vol. 230, No. 20, 2011, pp. 7670–7686. https://doi.org/10.1016/j.jcp.2011.06.023
[33] Katz, A. and Sankaran, V., "An Efficient Correction Method to Obtain a Formally Third-Order Accurate Flow Solver for Node-Centered Unstructured Grids," *Journal on Scientific Computing*, Vol. 51, 2012, pp. 375-393. https://doi.org/10.1007/s10915-011-9515-1.
[34] Liu, Y., and Nishikawa, H., "Third-Order Inviscid and Second-Order Hyperbolic Navier-Stokes Solvers for Three-Dimensional Inviscid and Viscous Flows," AIAA 2016-3969. https://doi.org/10.2514/6.2016-3969
[35] Liu, Y., and Nishikawa, H., "Third-Order Inviscid and Second-Order Hyperbolic Navier-Stokes Solvers for Three-Dimensional Unsteady Inviscid and Viscous Flows," AIAA 2017-0738. https://doi.org/10.2514/6.2017-0738
[36] Barth, T. J., "Numerical Aspects of Computing Viscous High Reynolds Number Flows on Unstructured Meshes," AIAA 1991-0721. https://doi.org/10.2514/6.1991-721
[37] Luo, H., Baum, J. D., Lohner, R., and Cabello, J., "Adaptive Edge-Based Finite Element Scheme for the Euler and Navier-Stokes Equations on Unstructured Grids," AIAA 93-0336. https://doi.org/10.2514/6.1993-336
[38] Mavriplis D.J., "Grid Resolution Study of a Drag Prediction Workshop Configuration Using the NSU3D Unstructured Mesh Solver," AIAA 2005-4729. https://doi.org/10.2514/6.2005-4729

[39] Nishikawa, H., "Two Ways to Extend Diffusion Scheme to Navier-Stokes Schemes: Gradient Formula or Upwind Flux," AIAA 2011-3044. https://doi.org/10.2514/6.2011-3044
[40] Tsieh, T., "An Investigation of Separated Flow About a Hemisphere Cylinder at 0- to 19-Deg Incidence in the Mach Number Range of 0.6 to 1.5," AEDC-TR-76-112, 1976.
[41] Rumsey, C. L., Carlson, J.-R., and Ahmad, N. N., "FUN3D Juncture Flow Computations Compared with Experimental Data," AIAA 2019-0079. https://doi.org/10.2514/6.2019-0079
[42] Kegerise, M. A. and Neuhart, D. H., "An Experimental Investigation of a Wing-Fuselage Junction Model in the NASA Langley 14- by 22-Foot Subsonic Tunnel," NASA/TM-2019–220286, NASA Langley Research Center, June 2019, Hampton, Virginia.
[43] Rumsey, C. L., Lee, H. C., and Pulliam, T. H., "Reynolds-Averaged Navier-Stokes Computations of the NASA Juncture Flow Model Using FUN3D and OVERFLOW," AIAA Paper 2020–1304. https://doi.org/10.2514/6.2020-1304
[44] Iyer, P. S. and Malik, M. R., "Wall-Modeled LES of the NASA Juncture Flow Experiment," AIAA Paper 2020–1307. https://doi.org/10.2514/6.2020-1307
[45] Diskin B., Ahmad N., Anderson W. K., Derlaga J. M., Pandya M. J., Rumsey C. L., Wang L., Wood S. L., Liu Y., Nishikawa H., and Galbraith M. C., "Verification Test Suite for Spalart-Allmaras QCR2000 Turbulence Model," AIAA 2021-1552. https://doi.org/10.2514/6.2021-1552
[46] Karman S. and Wyman N., "Automatic Unstructured Mesh Generation with Geometry Attribution," AIAA Paper 2019-1721. https://doi.org/10.2514/6.2019-1721