Introduction to COFFE: The Next-Generation HPCMP CREATE™-AV CFD Solver

Ryan S. Glasby[*], J. Taylor Erwin[†], and Douglas L. Stefanski[‡]

Joint Institute for Computational Sciences, University of Tennessee, Oak Ridge, TN 37831

Steven R. Allmaras[§] and Marshall C. Galbraith[¶]

Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, MA 02139

W. Kyle Anderson[‖]

NASA Langley Research Center, Hampton, VA 23681

Robert H. Nichols[**]

DoD HPCMP CREATE™-AVKetrel Team, Arnold AFB, TN 37389

HPCMP CREATE™-AV *Conservative Field Finite Element* (COFFE) is a modular, extensible, robust numerical solver for the Navier-Stokes equations that invokes modularity and extensibility from its first principles. COFFE implores a flexible, class-based hierarchy that provides a modular approach consisting of discretization, physics, parallelization, and linear algebra components. These components are developed with modern software engineering principles to ensure ease of uptake from a user's or developer's perspective. The Streamwise Upwind/Petrov-Galerkin (SU/PG) method is utilized to discretize the compressible Reynolds-Averaged Navier-Stokes (RANS) equations tightly coupled with a variety of turbulence models. The mathematics and the philosophy of the methodology that makes up COFFE are presented.

I. Introduction

The HPCMP Computational Research and Engineering for Acquisition Tools and Environments - Air Vehicles (CREATE™-AV) Program is developing a next-generation computational fluid dynamics (CFD) solver to address the need to be extensible within the same codebase with respect to spatial and temporal order of accuracy, available physics, and modern compute architectures. The SU/PG[1–8] finite-element method is employed to spatially discretize the governing equations, which allows a straightforward path to higher-order accuracy through the usage of hierarchical basis functions. The physics capability encompasses the time derivative, advection terms, diffusion terms, and source terms. Necessary physics capability is determined by the weighted residual formulation within the integrand of the finite-element method, which is written and solved for in conservative form by assumption. The parallelization strategy is to initially allow for distributed memory parallelization with message passing, while providing flexibility for emerging, hybrid architectures with respect to shared/distributed memory parallelization.

The software design philosophy for COFFE promotes flexibility, extensibility, and robustness through the utilization of modular software components coupled with functional capability quality assurance checks at compile time. In order to achieve a high level of flexibility and extensibility, each of the foundational components (discretization, physics, parallelization, and linear algebra) is implemented as an orthogonal module. To that end, component functionality is accessed through high-level, well-defined interfaces. Changes to underlying data structures and modification of implementation details require intra-component code changes only, while allowing inter-component and higher level

[*] Research Engineer, AIAA Member

[†] Research Engineer, AIAA Member

[‡] Research Engineer, AIAA Member

[§] Research Engineer, AIAA Associate Fellow

[¶] Research Engineer, AIAA Member

[‖] Research Engineer, AIAA Member

[**] Research Professor, University of Alabama at Birmingham, AIAA Associate Fellow

algorithms to remain unchanged. This compartmentalization is achieved through object-oriented and template techniques available in C++.[9] The cost reduction for software development arises naturally from the efficiences realized from the rigid, modern software engineering practices employed.

With the goal of providing confidence in the correctness and robustness of all source code within COFFE, extensive unit testing is required. Here, the Google Test[10] open source C++ testing framework is utilized. At least one unit test is created for every functional unit of source code. Upon compilation of the source code, all unit tests with modified dependancies are executed. A unit test failure at this stage constitutes a compilation failure, forcing the developer to examine the modified code. The entire suite of unit tests is run periodically through the Valgrind[11] dynamic memory analysis tool. In addition, the test suite is instrumented for code coverage as provided by the GNU C++[12] compiler. Coverage results are presented through a web interface by the open-source tool LCOV.[13]

The development of the COFFE codebase is user-centric, and COFFE is designed for ease of uptake and utilization. Software requirements from the user's perspective play an important role for the generation of COFFE's software architecture. COFFE is structured such that users can expect: (1) a hightly accurate answer that satisfies the governing nonlinear partial differential equations with wall-clock-time efficiency, (2) an intuitive grid-generation process, (3) a clear problem specification option through a user interface, (4) a production-ready CFD solver offering a new level of automation (potentially non-monitored code execution) and robustness, and (5) the availability of the appropriate physics for a given application.

The mitigation of current legacy issues and the improvement of Department of Defense air vehicle acquisition systems capability are the most relevant core requirements. The costly drawbacks of legacy CFD codes include: (1) originally written with non-modular software development paradigm, (2) the use of partially implicit and/or explicit temporal discretizations, (3) non-robust non-linear solution generation methods, (4) difficulties with the extension to various systems of equations, (5) difficulty of inclusion of various physical models, and (6) a lack of spatial/temporal higher-order accuracy. COFFE offers a counterpoint to each of these unresolved issues. The software design philosophy is fundamentally modular, is fully implicit in time, utilizes a robust exact linearization method, employs extensible equation systems, enables capacity for additional physical models, is higher-order accurate, and provides extensibility to error analysis and shape design through the usage of sensitivity derivatives computed via the discrete adjoint method. Overcoming these legacy CFD issues allows for enhanced air vehicle acquisition systems proficiency through enhanced flow-field analysis capability provided by the COFFE CFD tool.

II. Technical Attributes of COFFE

Technical aspects of the COFFE simulation tool include the governing equations, foundational solution framework, underlying software design methodology, parallelization for high-performance architectures, and comprehensive documentation. Subsequent subsections are ordered as follows: Section II.A details the Reynolds Averaged Navier-Stokes equations coupled with the Spalart-Allmaras[14, 15] (SA) turbulence model; Section II.B describes the necessary solution components of discretization, physics, parallelization, and linear algebra; Section II.C details the robust nonlinear solution path.

A. Governing Equations

The conservative form of the compressible Reynolds Averaged Navier-Stokes (RANS) equations, describing the conservation of mass, momentum, and total energy in three spatial dimensions tightly coupled with the SA turbulence model, are given as

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot \left(\mathbf{F}_c(\mathbf{u}) - \mathbf{F}_v(\mathbf{u}, \nabla \mathbf{u}) - \mathbf{F}_{ad}(\mathbf{u}, \nabla \mathbf{u}) \right) = \mathbf{S}(\mathbf{u}, \nabla \mathbf{u}) \tag{1}$$

$$\mathbf{u} = \begin{Bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho e_t \\ \rho \tilde{\nu} \end{Bmatrix} \tag{2}$$

$$\mathbf{F}_c^x = \begin{Bmatrix} \rho u \\ \rho u^2 + p \\ \rho u v \\ \rho u w \\ (\rho e_t + p) u \\ \rho \tilde{\nu} u \end{Bmatrix} \quad \mathbf{F}_c^y = \begin{Bmatrix} \rho v \\ \rho u v \\ \rho v^2 + p \\ \rho v w \\ (\rho e_t + p) v \\ \rho \tilde{\nu} v \end{Bmatrix} \quad \mathbf{F}_c^z = \begin{Bmatrix} \rho w \\ \rho u v \\ \rho v v \\ \rho w^2 + p \\ (\rho e_t + p) w \\ \rho \tilde{\nu} w \end{Bmatrix} \tag{3}$$

$$\mathbf{F}_v^x = \begin{Bmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ \tau_{xz} \\ u\tau_{xx} + v\tau_{xy} + w\tau_{xz} - q_x \\ \frac{1}{\sigma}(\mu + f_n \rho \tilde{\nu}) \frac{\partial \tilde{\nu}}{\partial x} \end{Bmatrix} \quad \mathbf{F}_v^y = \begin{Bmatrix} 0 \\ \tau_{yx} \\ \tau_{yy} \\ \tau_{yz} \\ u\tau_{yx} + v\tau_{yy} + w\tau_{yz} - q_y \\ \frac{1}{\sigma}(\mu + f_n \rho \tilde{\nu}) \frac{\partial \tilde{\nu}}{\partial y} \end{Bmatrix} \quad \mathbf{F}_v^z = \begin{Bmatrix} 0 \\ \tau_{zx} \\ \tau_{zy} \\ \tau_{zz} \\ u\tau_{zx} + v\tau_{zy} + w\tau_{zz} - q_z \\ \frac{1}{\sigma}(\mu + f_n \rho \tilde{\nu}) \frac{\partial \tilde{\nu}}{\partial z} \end{Bmatrix} \tag{4}$$

$$\mathbf{F}_{ad}^x = \begin{Bmatrix} h\lambda_{\max}\epsilon_{\text{shock}} \frac{\partial \rho}{\partial x} \\ h\lambda_{\max}\epsilon_{\text{shock}} \frac{\partial \rho \mathbf{u}}{\partial x} \\ h\lambda_{\max}\epsilon_{\text{shock}} \frac{\partial \rho \mathbf{v}}{\partial x} \\ h\lambda_{\max}\epsilon_{\text{shock}} \frac{\partial \rho \mathbf{w}}{\partial x} \\ h\lambda_{\max}\epsilon_{\text{shock}} \frac{\partial \rho \mathbf{h_t}}{\partial x} \\ 0.0 \end{Bmatrix} \quad \mathbf{F}_{ad}^y = \begin{Bmatrix} h\lambda_{\max}\epsilon_{\text{shock}} \frac{\partial \rho}{\partial y} \\ h\lambda_{\max}\epsilon_{\text{shock}} \frac{\partial \rho \mathbf{u}}{\partial y} \\ h\lambda_{\max}\epsilon_{\text{shock}} \frac{\partial \rho \mathbf{v}}{\partial y} \\ h\lambda_{\max}\epsilon_{\text{shock}} \frac{\partial \rho \mathbf{w}}{\partial y} \\ h\lambda_{\max}\epsilon_{\text{shock}} \frac{\partial \rho \mathbf{h_t}}{\partial y} \\ 0.0 \end{Bmatrix} \quad \mathbf{F}_{ad}^z = \begin{Bmatrix} h\lambda_{\max}\epsilon_{\text{shock}} \frac{\partial \rho}{\partial z} \\ h\lambda_{\max}\epsilon_{\text{shock}} \frac{\partial \rho \mathbf{u}}{\partial z} \\ h\lambda_{\max}\epsilon_{\text{shock}} \frac{\partial \rho \mathbf{v}}{\partial z} \\ h\lambda_{\max}\epsilon_{\text{shock}} \frac{\partial \rho \mathbf{w}}{\partial z} \\ h\lambda_{\max}\epsilon_{\text{shock}} \frac{\partial \rho \mathbf{h_t}}{\partial z} \\ 0.0 \end{Bmatrix} \tag{5}$$

$$\mathbf{S} = \begin{Bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \frac{c_{b2}}{\sigma} \rho (\nabla \tilde{\nu} \cdot \nabla \tilde{\nu}) - \frac{1}{\sigma}(\nu + \tilde{\nu})(\nabla \rho \cdot \nabla \tilde{\nu}) + \rho P_n - \rho D_n - C_5 \frac{\rho \tilde{\nu}^2}{a^2} \frac{\partial u_i}{\partial x_j} \frac{\partial u_i}{\partial x_j} \end{Bmatrix} \tag{6}$$

where ρ is fluid density, $\mathbf{u} = (u, v, w)$ are the Cartesian velocity components, p is the fluid pressure, e_t is the total energy per unit mass, τ_{ij} is the total viscous stress tensor including the Boussinesq approximated Reynolds stresses, and $\mathbf{q} = (q_x, q_y, q_z) = -\kappa \nabla \mathbf{T}$ are the Cartesian components of heat conduction. Assuming a Newtonian fluid and using the Boussinesq approximation for Reynolds stresses, the viscous stress tensor takes the form

$$\begin{aligned} \tau_{xx} &= (\mu + \mu_T)\left(\frac{4}{3}\frac{\partial u}{\partial x} - \frac{2}{3}\left(\frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}\right)\right) & \tau_{xy} &= \tau_{yx} = (\mu + \mu_T)\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right) \\ \tau_{yy} &= (\mu + \mu_T)\left(\frac{4}{3}\frac{\partial u}{\partial y} - \frac{2}{3}\left(\frac{\partial u}{\partial x} + \frac{\partial w}{\partial z}\right)\right) & \tau_{xz} &= \tau_{zx} = (\mu + \mu_T)\left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x}\right) \\ \tau_{zz} &= (\mu + \mu_T)\left(\frac{4}{3}\frac{\partial u}{\partial z} - \frac{2}{3}\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right)\right) & \tau_{yz} &= \tau_{zy} = (\mu + \mu_T)\left(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y}\right) \end{aligned} \tag{7}$$

where μ is the fluid viscosity obtained via Sutherland's law, ν is the kinematic viscosity of the fluid, and μ_T is a turbulent eddy viscosity, which is given by

$$\mu_T = \begin{cases} \rho \tilde{\nu} f_{v_1} & \tilde{\nu} \geq 0 \\ 0 & \tilde{\nu} < 0 \end{cases} \tag{8}$$

$$f_{v_1} = \frac{\chi^3}{\chi^3 + c_{v_1}^3} \qquad \chi \equiv \frac{\tilde{\nu}}{\nu} \tag{9}$$

where $\tilde{\nu}$ is the SA model working variable, $c_{v_1} = 7.1$, $c_{b2} = 0.622$, and $\sigma = 2/3$. The following terms pertaining to the SA turbulence model are defined as

$$f_n = \begin{cases} 1 & \tilde{\nu} \geq 0 \\ \dfrac{c_{n1} + \chi^3}{c_{n1} - \chi^3} & \tilde{\nu} < 0 \end{cases} \tag{10}$$

$$P_n = \begin{cases} c_{b1}(1 - f_{t_2})\tilde{S}\tilde{\nu} & \tilde{\nu} \geq 0 \\ c_{b1}(1 - c_{t_3})\tilde{S}\tilde{\nu} & \tilde{\nu} < 0 \end{cases} \tag{11}$$

$$\tilde{S} = \begin{cases} S + \bar{S} & \bar{S} > -c_{v_2}S \\ S + \dfrac{S(c_{v_2}^2 S + c_{v_3}\bar{S})}{(c_{v_3} - 2c_{v_2})S - \bar{S}} & \bar{S} \leq -c_{v_2}S \end{cases} \tag{12}$$

$$S = \nabla \times \mathbf{u} = \sqrt{\left(\frac{\partial w}{\partial y} - \frac{\partial v}{\partial z}\right)^2 + \left(\frac{\partial u}{\partial z} - \frac{\partial w}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}\right)^2} \qquad \bar{S} = \frac{\tilde{\nu} f_{v_2}}{\kappa_{\mathrm{SA}}^2 d^2} \tag{13}$$

$$D_n = \begin{cases} \left[c_{w1} f_w - \dfrac{c_{b1}}{\kappa_{\mathrm{SA}}^2} f_{t_2}\right] \left(\dfrac{\tilde{\nu}}{d}\right)^2 & \tilde{\nu} \geq 0 \\ -c_{w1} \left(\dfrac{\tilde{\nu}}{d}\right)^2 & \tilde{\nu} < 0 \end{cases} \tag{14}$$

$$C_5 = 3.5\left(1.0 - \tanh\left(\left[\frac{24.0\tilde{\nu}}{\sqrt{\frac{\partial u_i}{\partial x_j}\frac{\partial u_i}{\partial x_j}}\kappa_{\mathrm{SA}}^2 d^2}\right]^3\right)\right) \tag{15}$$

where d is the distance to the nearest wall, $c_{n1} = 16.0$, $c_{b1} = 0.1355$, the laminar suppression term $f_{t_2} = c_{t_3}\exp(-c_{t_4}\chi^2)$, and C_5[16] is multiplied by a slightly modified DDES[17] term (f_d). The following SA terms are defined as

$$\begin{aligned} f_{v_2} &= 1 - \frac{\chi}{1 + \chi f_{v_1}} \qquad & f_w &= g\left(\frac{1 + c_{w3}^6}{g^6 + c_{w3}^6}\right)^{1/6} \\ g &= r + c_{w2}(r^6 - r) \qquad & r &= \min\left(\frac{\tilde{\nu}}{\tilde{S}\kappa_{\mathrm{SA}}^2 d^2}, r_{\mathrm{lim}}\right) \end{aligned} \tag{16}$$

The constants are $\kappa_{\mathrm{SA}} = 0.41$, $c_{w_1} = \frac{c_{b_1}}{\kappa_{\mathrm{SA}}^2} + \frac{1+c_{b_2}}{\sigma}$, $c_{w_2} = 0.3$, $c_{w_3} = 2.0$, $c_{v_2} = 0.7$, $c_{v_3} = 0.9$, $c_{t_3} = 1.2$, $c_{t_4} = 0.5$, and $r_{\mathrm{lim}} = 10.0$. All quantities in the above equations are Reynolds averaged. Initially, pressure is obtained from the ideal gas equation of state, but other real gas options will be available. The ideal gas equation of state is specified as

$$p = (\gamma - 1)\left(\rho e_t - \frac{1}{2}\rho\left(u^2 + v^2 + w^2\right)\right) \tag{17}$$

where γ is the ratio of specific heats. With respect to the artificial diffusion flux, h is an element length measured as area divided by perimeter in two dimensions and volume divided by the sum of the face areas in three dimensions. λ_{max} is defined as

$$\lambda_{\mathrm{max}} = |u| + |v| + |w| + c \tag{18}$$

$\epsilon_{\mathrm{shock}}$ is defined as

$$\epsilon_{\mathrm{shock}} = \frac{|\nabla \cdot p|\, h}{|\nabla \cdot p|\, h + p} \tag{19}$$

The governing equations are identical for dimensional or non-dimensional applications as long as the flow variables are non-dimensionalized in the following way (non-dimensional variables denoted by *):

$$\begin{array}{ccccc} x^* = \dfrac{x}{L} & u^* = \dfrac{u}{a_\infty} & \rho^* = \dfrac{\rho}{\rho_\infty} & t^* = \dfrac{ta_\infty}{L} & p^* = \dfrac{p}{\rho_\infty a_\infty^2} \\ T^* = \dfrac{T}{T_\infty} & e_t^* = \dfrac{e_t}{a_\infty^2} & \mu^* = \dfrac{\mu}{\rho_\infty a_\infty L} & \kappa^* = \dfrac{\kappa T_\infty}{\rho a_\infty^3 L} & \tilde{\nu}^* = \dfrac{\tilde{\nu}}{a_\infty L} \end{array} \tag{20}$$

B. Foundational Solution Computation Components

The foundational solution components are the framework for the delivery of a successful, extensible, and robust solution method. This section gives a brief overview of each of these foundational aspects. In the subsections that follow, increased depth is put forth for each facet of the computed solution algorithm. The specifics of the technical approach rest on the core components of spatial discretization, physics, parallelization, and linear algebra. A continuous finite element method is utilized to spatially discretize the governing equations in a performance-efficient and robust manner. The physics components are implemented in an extensible manner by allowing availability of multiple sets of partial differential equations and boundary conditions within a single codebase. The parallelization component employs the message passing interface[18] (MPI) library to conduct asynchronous communication that allows data to be sent across a distributed memory supercomputer concurrent to other computational operations, mitigating data latency. In addition to efficient data sending/receiving, an advanced threading model will ensure that on-node parallelism will be exploited effectively and efficiently. The linear algebra component consists of multiple linear solvers and matrix pre-conditioners. This approach allows for the utilization of the most efficient and robust linear solver and pre-conditioner for a given application, which maps the best linear solver to the current problem. In essence, the goal of this software architecture is to provide flexibility and extensibility (with respect to the core components) while providing extreme robustness without sacrificing performance.

1. Spatial Discretization using Continuous Finite Element Methods

A continuous Galerkin stabilized finite element method is employed to discretize the governing equations. Finite element methods in general have many advantages over other discretization schemes. Specifically, they (1) do not employ solution gradient reconstruction to attain higher-order accuracy, (2) are more robust on distorted grids than gradient reconstruction finite volume methods with respect to accuracy, (3) utilize a compact (nearest neighbor) stencil, (4) are capable of being linearized even for higher order spatial accuracy, (5) are built on a rigorous mathematical framework, (6) have well defined boundary conditions, (7) have a straightforward path to higher-order accuracy, (8) are adjoint consistent, (9) are subject to minimal overlap requirements for overset grid methodology, and (10) are subject to minimal parallel communication overhead. The authors have extensive experience[8, 19–22] with implementing the Streamline Upwind/Petrov-Galerkin (SU/PG) method. Currently, within COFFE, the SU/PG method, the Galerkin Least Squares[23] or the Variational Multiscale[24] finite element methods are implemented. All three methods are similar and are identical for the Euler equations. Each of the methods stores the solution data at the vertices of the mesh, and each method incorporates upwinding through modification of the basis function. A derivation of the SU/PG method utilized in COFFE for the Navier-Stokes equations is shown as follows (starting with a restatement of the governing equations in conservative form previously given as Eq. (1)):

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot (\mathbf{F}_c(\mathbf{u}) - \mathbf{F}_v(\mathbf{u}, \nabla\mathbf{u}) - \mathbf{F}_{ad}(\mathbf{u}, \nabla\mathbf{u})) = \mathbf{S}(\mathbf{u}, \nabla\mathbf{u}) \tag{21}$$

The SU/PG method starts by introducing an approximate solution $\mathbf{u}_h \in V_h^p$ as

$$\mathbf{u}_h = \sum_i \hat{\mathbf{u}}_i \phi_i(\mathbf{x}) \tag{22}$$

where $\phi_i(\mathbf{x})$ are known as basis functions. Substituting $\mathbf{u}_h$ for $\mathbf{u}$ into Eq. (21) yields

$$\frac{\partial \mathbf{u}_h}{\partial t} + \nabla \cdot (\mathbf{F}_c(\mathbf{u}_h) - \mathbf{F}_v(\mathbf{u}_h, \nabla\mathbf{u}_h) - \mathbf{F}_{ad}(\mathbf{u}_h, \nabla\mathbf{u}_h)) - \mathbf{S}(\mathbf{u}_h, \nabla\mathbf{u}_h) \neq 0 \tag{23}$$

since the approximate solution $\mathbf{u}_h$ does not exactly satisfy the governing equations. The functions $\phi_i(\mathbf{x})$ are known; and therefore, the solution to the approximate problem is found if one can find suitable values for the unknown coefficients.

Galerkin methods find the unknown coefficients via a weighted residual approach by taking an inner product between Eq. (23) and a set of known functions $\mathbf{v}_h$:

$$\sum_{k\in\mathcal{T}_h} \oiiint \mathbf{v}_h^{\mathsf{T}} \frac{\partial \mathbf{u}_h}{\partial t} + \mathbf{v}_h^{\mathsf{T}} \nabla \cdot \left(\mathbf{F}_c(\mathbf{u}_h) - \mathbf{F}_v(\mathbf{u}_h, \nabla \mathbf{u}_h) - \mathbf{F}_{ad}(\mathbf{u}_h, \nabla \mathbf{u}_h)\right) - \mathbf{v}_h^{\mathsf{T}} \mathbf{S}(\mathbf{u}_h, \nabla \mathbf{u}_h)\, d\Omega_k = 0 \tag{24}$$

$\forall \mathbf{v}_h \in V_h^p$. The SU/PG method assumes continuity of the basis functions between elements and sets the function $\mathbf{v}_h$ to be

$$\mathbf{v}_h = \phi_i + \left[\frac{\phi_i}{\partial t} + \nabla \phi_i \cdot \frac{\partial \mathbf{F}_c(\mathbf{u}_h)}{\partial \mathbf{u}_h} - \phi_i \frac{\partial \mathbf{S}(\mathbf{u}_h, \nabla \mathbf{u}_h)}{\partial \mathbf{u}_h}\right] [\boldsymbol{\tau}], \quad \forall i \tag{25}$$

Substituting these values into Eq. (24) and integrating by parts once gives

$$\begin{aligned}
&\sum_{e\in\mathcal{T}_h} \oiiint \phi_i \frac{\partial \mathbf{u}_h}{\partial t}\, d\Omega_e - \sum_{e\in\mathcal{T}_h} \oiiint \left[\nabla \phi_i \cdot (\mathbf{F}_c(\mathbf{u}_h) - \mathbf{F}_v(\mathbf{u}_h, \nabla \mathbf{u}_h) - \mathbf{F}_{ad}(\mathbf{u}_h, \nabla \mathbf{u}_h)) + \phi_i \mathbf{S}(\mathbf{u}_h, \nabla \mathbf{u}_h)\right] d\Omega_e \\
&+ \sum_{e\in\mathcal{T}_h} \oiiint \left[\frac{\phi_i}{\partial t} + \nabla \phi_i \cdot \frac{\partial \mathbf{F}_c(\mathbf{u}_h)}{\partial \mathbf{u}_h} - \phi_i \frac{\partial \mathbf{S}(\mathbf{u}_h, \nabla \mathbf{u}_h)}{\partial \mathbf{u}_h}\right] [\boldsymbol{\tau}] \left[\frac{\partial \mathbf{u}_h}{\partial t} + \nabla \cdot \left(\mathbf{F}_c(\mathbf{u}_h) - \mathbf{F}_v(\mathbf{u}_h, \nabla \mathbf{u}_h)\right) - \mathbf{S}(\mathbf{u}_h, \nabla \mathbf{u}_h)\right] d\Omega_e \\
&+ \sum_{e\in\mathcal{T}_h} \oiint \phi_i \mathbf{n} \cdot \left(\mathbf{F}_c(\mathbf{u}_b) - \mathbf{F}_v((\mathbf{u}_h, \mathbf{u}_b), \nabla \mathbf{u}_h)\right) d\Gamma^b = 0
\end{aligned} \tag{26}$$

The $[\boldsymbol{\tau}]$ matrix is given as

$$[\boldsymbol{\tau}]^{-1} = \sum_i \left(\frac{\phi_i}{\partial t} + \left|\nabla \phi_i \cdot \frac{\partial \mathbf{F}_c(\mathbf{u}_h)}{\partial \mathbf{u}_h}\right| + \nabla \phi_i \cdot \frac{\partial \mathbf{F}_v(\mathbf{u}_h, \nabla \mathbf{u}_h)}{\partial (\nabla \mathbf{u}_h)} \cdot \nabla \phi_i - \phi_i \frac{\partial \mathbf{S}(\mathbf{u}_h, \nabla \mathbf{u}_h)}{\partial \mathbf{u}_h}\right) \tag{27}$$

A critical component of any SU/PG finite-element method is the form of

$$\sum_i \left|\nabla \phi_i \cdot \frac{\partial \mathbf{F}_c(\mathbf{u}_h)}{\partial \mathbf{u}_h}\right| \tag{28}$$

which is taken as the following dissipation matrix for COFFE:

$$\sum_i \left|\nabla \phi_i \cdot \frac{\partial \mathbf{F}_c(\mathbf{u}_h)}{\partial \mathbf{u}_h}\right| = [\boldsymbol{T}][|\boldsymbol{\Lambda}|][\boldsymbol{T}]^{-1} \tag{29}$$

where $[\boldsymbol{T}]$ is the matrix of eigenvectors and $[|\boldsymbol{\Lambda}|]$ is the matrix of eigenvalues of the matrix in Eq. (29). The viscous boundary flux is evaluated via the symmetric interior penalty method[25]

$$\mathbf{n} \cdot \mathbf{F}_v((\mathbf{u}_h, \mathbf{u}_b), \nabla \mathbf{u}_h) = \mathbf{n} \cdot \mathbf{F}_v(\mathbf{u}_b, \nabla \mathbf{u}_h) + \nabla \phi_i \cdot \mathbf{F}_v(\mathbf{u}_b, \mathbf{n} \cdot (\mathbf{u}_h - \mathbf{u}_b)) - \mathbf{n} \cdot \mathbf{F}_v(\mathbf{u}_b, \delta \mathbf{n} \cdot (\mathbf{u}_h - \mathbf{u}_b)) \tag{30}$$

where $\delta = \frac{\#Basis_h}{h_h}$. h_h is defined as

$$h_h = \frac{1.0}{\sum_i \sqrt{\nabla \phi_i \cdot \nabla \phi_i}} \tag{31}$$

2. *Multi-Physics Capability*

The physics capability encompasses the time derivative, advection terms, diffusion terms, and source terms, and is determined by the governing equations within the integrand of the finite element method, which are written in conservative form by assumption. The physics capability is written in an extensible manner, incorporating the correct physics/governing equations for the particular application.

A robust steady -state CFD tool must appropriately couple the RANS equations with a variety of turbulence models and gas assumptions, provide for appropriate boundary conditions, and allow for tight/loose coupling with external physics capabilities. Initially, the RANS equations with the SA turbulence model and ideal gas assumption are implemented to allow modeling of steady/unsteady sub/super-sonic flow. Additionally, RANS equations with various two-equation turbulence models will be implemented as well as real gas and finite-rate chemistry interactions that are appropriate for combustion and hypersonic flow. Boundary condition types that are available include: no-slip adiabatic or constant temperature wall, slip wall, source/sink, and subsonic/supersonic in/out flow characteristics. The boundary conditions are implemented in a weak fashion to allow for adjoint consistency. The equation sets are fully coupled with respect to the flow/turbulence variables. COFFE will also support loose coupling with other non-native physics capabilities (structural modeling, electromagnetic field modeling, etc.) within the Newton step of the non-linear solution scheme.

3. *Parallelization Strategy*

Modern high-performance computing architectures rely heavily on increasing numbers of parallel computing cores to provide increasing aggregate computational performance, both within conventional CPUs and within supplementary devices. This trend is expected to continue over the next several generations of emerging computing architectures, possibly leading to systems with hundreds of cores per node and millions of cores in aggregate being widely available in the near future. Such rapidly increasing parallelism at both the node level and the machine level places significant burdens on developers of scientific and engineering software that seek to effectively and efficiently use modern, high-performance supercomputing platforms.

With modern, high-performance supercomputing plaforms in mind, the parallelization strategy for COFFE is practical and extensible. The strategy is to initially allow for distributed memory parallelization with MPI, while providing flexibility for emerging, hybrid architectures. Thereby, code longevity is ensured relative to future computing platforms. The current strategy for scientific computing readiness is an implementation of MPI-level parallelism on a distributed memory supercomputer. Effective mesh partitioning, communication lists, and a clever nodal arrangement allowd for concurrent work division and a non- blocking communication model. Domain decomposition of underlying, unstructured meshes is carried out using Zoltan.[26] The input to Zoltan is a parallel node adjacency map that is built off a mesh element connectivity array. Send/receive lists are used to allow for non-blocking MPI library send/receive calls that dictate source/destination of communicated data. This approach, coupled with the concise stencil of a continuous finite element method, brings coarse-grain parallel efficiency to the COFFE codebase. The long-term strategy for COFFE focuses on future development efforts to implement intra-node threading models to support mid-grain decomposition and task-based parallelism, while exposing fine-grain parallelism suitable for vector processors. The resulting hierarchical model for parallelism will enable the adaptation of the COFFE codebase to expected future architectures.

4. *Linear Algebra*

The linear algebra component of COFFE is extremely important. Due to the choice of an implicit methodology, an update to the solution vector, via a linear system solve, must be computed at each time step. High performing, robust linear solvers and matrix pre-conditioners represent active areas of research. To ensure the availability of a wide range of current and emerging algorithms, linear algebra in COFFE is accessed through a high-level interface behind which implementation details are hidden. This level of flexibility allows COFFE developers to choose linear solvers and matrix pre- conditioners based on the problem being solved without modification of other core components. The core linear algebra solvers that are available are Generalized Minimum Residual (GMRES),[27] Gauss-Seidel,[28] and point Jacobi.[28] Incomplete LU[27] decomposition is available as a matrix pre-conditioner for the GMRES linear algebra solver. In the first quarter of 2016, an unstructured line Gauss-Seidel solver will be available as a stand alone solver and as a preconditioner. This solver draws lines in a novel way through an unstructured mesh by evaluating the strength of connection[29] of the linearization matrix of Laplace's equation applied to the unstructured mesh. Finally, by utilizing an exact linearization of the residual, a true Newton's method is obtained. The exact linearization of the residual is guarenteed through the usage of operator overloaded variables[30] to calculate the residual vector. A notable side effect of this choice is the availability of the linearization required for adjoint based error estimation and shape- design/optimization.

C. Nonlinear Path to Steady-State Convergence

One of the most critical and time-consuming aspects of a CFD solver is the resolution of the non-linear system of algebraic equations that result from spatial and temporal discretization. One of the CREATE™-AVprogram's main goals to to develop highly automated and fast CFD solvers. Therefore, an implicit solution method,[31] which gives fast solution convergence for steady-state flows and allows user complete freedom in setting the time-step for time-accurate flows, is a design requirement. COFFE has the ability to efficiently/automatically traverse the nonlinear path to steady-state convergence. This is accomplished by first applying Newton's method to the pseudo-transient residual

$$0 = \mathbf{R}^{n+1} = \mathbf{R}^{n} + \frac{\partial \mathbf{R}^{n}}{\partial \mathbf{u}^{n}} \Delta \mathbf{u}^{n+1} \tag{32}$$

The time derivative within $\mathbf{R}^n$ is approximated locally as

$$\partial t_h = CFL \frac{h_h}{c_h + |u_h| + |v_h| + |w_h|} \tag{33}$$

where c is the local speed of sound. At every Newton step the update to the solution vector $\Delta\mathbf{u}$ is solved for via

$$\frac{\partial \mathbf{R}^n}{\partial \mathbf{u}^n} \Delta\mathbf{u}^{n+1} = -\mathbf{R}^n \tag{34}$$

An under-relaxation parameter, ω^n, is applied to the computed update to ensure a robust path to convergence, and the solution at the next pseudo time step is

$$\mathbf{u}^{n+1} = \mathbf{u}^n + \omega^n \Delta\mathbf{u}^{n+1} \tag{35}$$

The value of ω^n is calculated via a two step process as

$$\left| \frac{\omega_\rho^n \Delta\rho^{n+1}}{\rho^n} \right|_\infty < \eta, \omega_\rho \in [0,1] \tag{36}$$

$$\left| \frac{\omega_T^n \Delta T^{n+1}}{T^n} \right|_\infty < \eta, \omega_T \in [0,1] \tag{37}$$

$$\omega_{physical}^n = \min(\omega_\rho^n, \omega_T^n) \tag{38}$$

Usually the value η is given as 0.30, and $\omega_{physical}^n$ is constant in space. The second method of choosing ω^n uses a globalization technique called a line search algorithm. The goal of the line search algorithm is to find an acceptable update where the pseudo-transient unsteady residual decreases. The line search method begins with the value of $\omega_{physical}^n$ and determines the final value of ω^n as

$$\left| \mathbf{R}(\mathbf{u}^n + \omega^n \Delta\mathbf{u}^{n+1}) \right|_2 - \alpha \left| \mathbf{R}(\mathbf{u}^n) \right|_2 < 0.0 \wedge \max \omega^n \tag{39}$$

where α is given as 1.5. The CFL is calculated based on ω^n as

$$CFL^n = \begin{cases} \frac{CFL^{n-1}}{10.0}, \omega^n < 0.1 \\ CFL^{n-1}, 0.1 < \omega^n < 1.0 \\ 2.0 \cdot CFL^{n-1}, \omega^n = 1.0 \end{cases} \tag{40}$$

III. Results

Since the codebase within COFFE is constantly growing, system tests are regularly operated to ensure COFFE retains the capability to accurately model the flow-field surrounding trans/super-sonic aircraft with a turbulence model. One system test utilizes a 2-D mesh of a NACA0012 airfoil made up of 35,600 nodes and 35,052 quadrilateral elements depicted in Figure 1, with four flow conditions and the following expected forces and number of Newton steps to drive the mass conservation residual below $1.0e-12$. Non-dimensional densitiy contours are depicted in Figures 2, 4, **??**, and 5.

Table 1. NACA 0012 - Simulation Parameters

Mach Number	**Angle of Attack**	**Reynolds # MAC**	C_L	C_D	**Newton Steps**
0.79	2.0	9.0e6	0.36545	0.031284	127
0.79	2.0	12.0e6	0.37120	0.031256	130
0.79	5.0	12.0e6	0.44131	0.065879	243
2.0	0.0	12.0e6	0.00100	0.095081	222

Another system test utilizes a 2-D mesh of a 30p30n airfoil made up of 42,337 nodes and 83,971 triangular elements depicted in Figure 6, with flow conditions and the following expected forces and number of Newton steps to drive the mass conservation residual below $1.0e-12$. Non-dimensional densitiy contours are depicted in Figure 7.

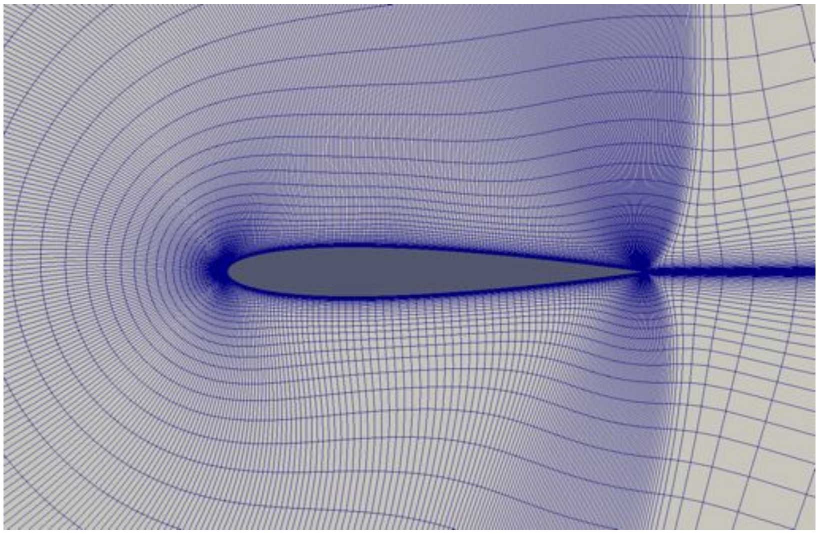

Figure 1. NACA 0012: Grid

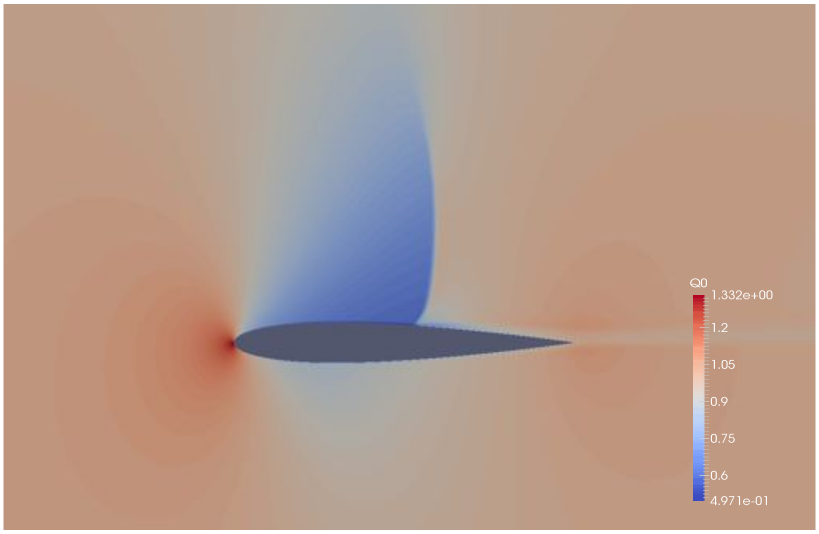

Figure 2. Non-dimensional Density Contours, Mach = 0.79, $\alpha = 2.0$, $Re_{MAC} = 9.0e6$

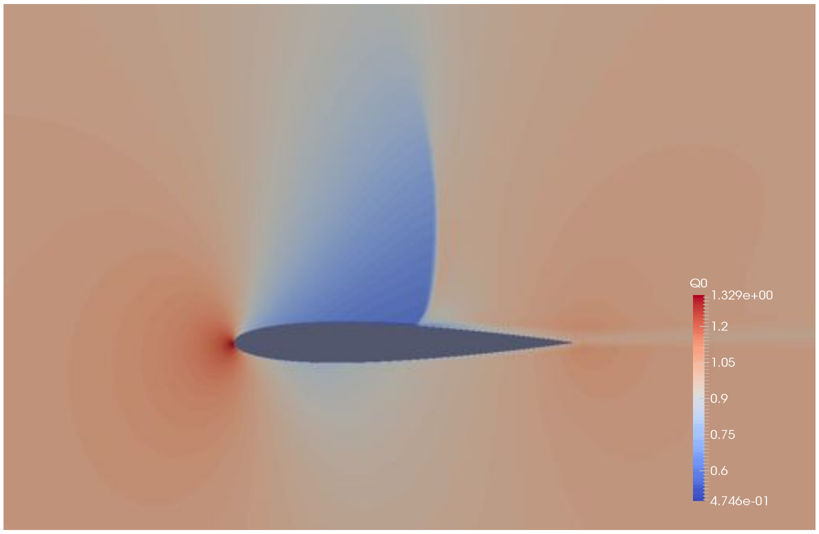

Figure 3. Non-dimensional Density Contours, Mach = 0.79, α = 2.0, Re_{MAC} = 12.0e6

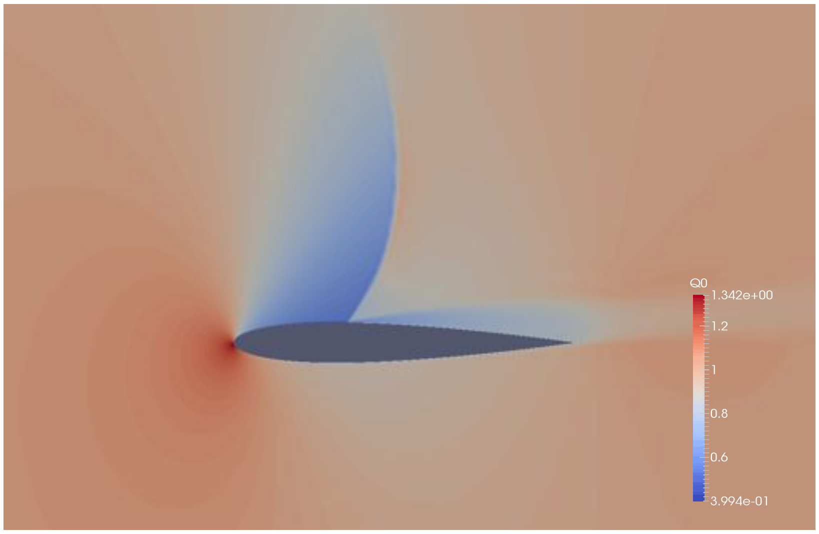

Figure 4. Non-dimensional Density Contours, Mach = 0.79, α = 5.0, Re_{MAC} = 12.0e6

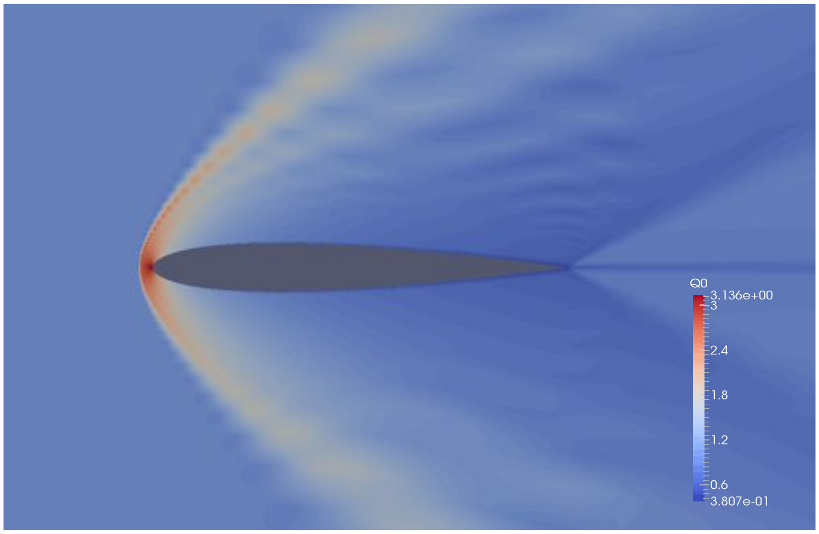

Figure 5. Non-dimensional Density Contours, Mach = 2.0, α = 0.0, Re_{MAC} = 12.0e6

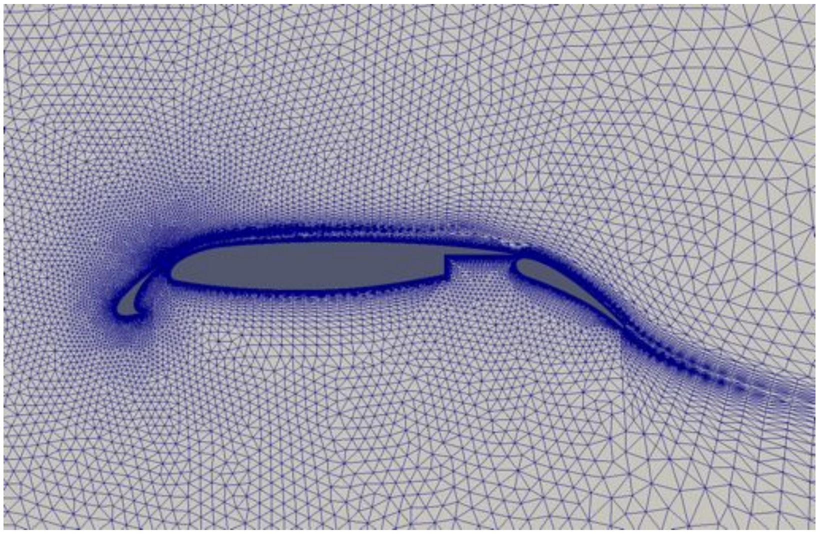

Figure 6. 30p30n: Grid

Table 2. 30p30n - Simulation Parameters

Mach Number	Angle of Attack	Reynolds # MAC	C_L	C_D	Newton Steps
0.20	16.0	9.0e6	4.14371	0.056386	339

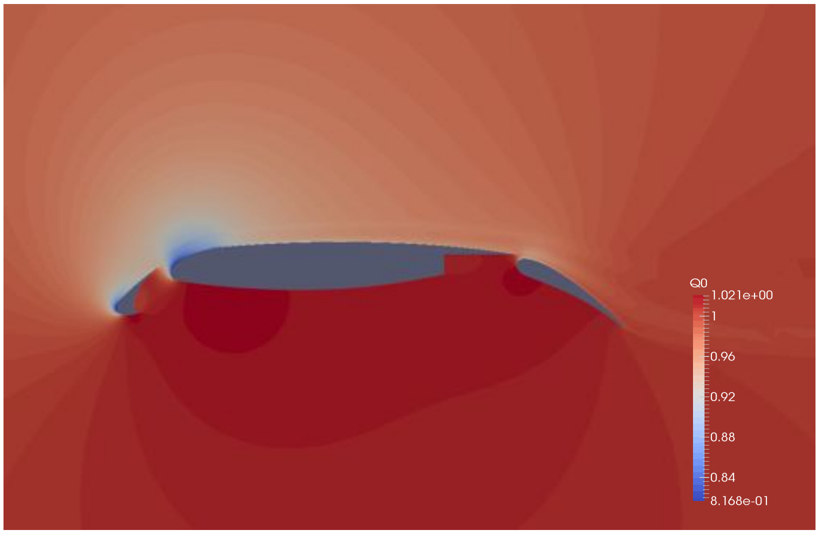

Figure 7. Non-dimensional Density Contours, Mach = 0.2, α = 16.0, Re_{MAC} = 9.0e6

IV. Conclusions

Addressing core requirements of reducing costs, reducing schedules, improving the capability of DoD air vehicle acquisition systems, and mitigating the drawbacks of off-the-shelf legacy CFD tools factor prominently into the crafting of underlying technical capabilities of COFFE. The choice of fundamental solver components, software design philosophy, and user-centric design all play integral roles in computing the solution to the Navier-Stokes equations while keeping the overall focus on the core program requirements.

The cost reduction for software development arises naturally from the efficiencies realized from the rigid, modern software engineering practices employed. The solver architecture for COFFE is designed in terms of modular capabilities implemented in an extensible manner using a templated, object-oriented, interface-driven software design approach. This simple paradigm promotes abstraction, masks low-level facilities, and greatly reduces the time required for development. Additionally, the continuous unit testing, code coverage verification, and memory leak detection dramatically reduce the likelihood of incorrect software implementations. Parallelization is addressed with extensible design patterns that are applicable for efficient computing on current HPCMP architectures as well as future HPCMP architecture. Ease of use is ensured through user-centric tools and practices. To address the current and future needs of the HPCMP program all aspects of the COFFE CFD simulation tool are instituted within a flexible, modular software architecture that is extensible with respect to the core components.

Acknowledgments

Material presented in this paper is a product of the Air Vehicles element of the HPCMP CREATE™ Program. The authors would like to thank the HPCMP for the many hours of computing time that are so critical for the successful development and validation of COFFE.

The authors would also like to thank their colleagues on the CREATE™-AV senior management and Kestrel teams for their support and hard work in producing the exceptional HPCMP CREATE™-AV Kestrel product. Specifically, a special thanks goes to Dr. Robert Meakin and Dr. Nathan Hariharan as well as Dr. Scott Morton and Dr. David McDaniel.

References

[1] Brooks, A. N. and Hughes, T. J. R., "Streamline Upwind/Petrov-Galerkin Formulations for Convection Dominated Flows with Particular Emphasis on the Incompressible Navier-Stokes Equations," *Computer Methods in Applied Mechanics and Engineering*, Vol. 32, No. 1–3, Sept. 1982, pp. 199–259.

[2] Barth, T. J., "Numerical methods for gasdynamic systems on unstructured meshes," *An Introduction to Recent Developments in Theory and Numerics for Conservation Laws*, edited by D. Kroner, M. Ohlberger, and C. Rhode, Vol. 5, Springer, New York, NY, USA, 1998, pp. 195–285.

[3] Bonhaus, D. L., *A Higher Order Accurate Finite Element Method for Viscous Compressible Flows*, Ph.D. thesis, Virginia Polytechnic Institute and State University, Nov. 1998.

[4] Franca, L. P., Frey, S. L., and Hughes, T. J. R., "Stabilized finite element methods: I. Application to the advective-diffusive model," *Computer Methods in Applied Mechanics and Engineering*, Vol. 95, No. 2, 1992, pp. 253–276.

[5] Hughes, T. J. R., "Recent progress in the development and understanding of SUPG methods with special reference to the compressible Euler and Navier-Stokes equations," *International Journal for Numerical Methods in Fluids*, Vol. 7, No. 11, 1987, pp. 1261–1275.

[6] Kirk, B. S. and Carey, G. F., "Development and validation of a SUPG finite element scheme for the compressible Navier-Stokes equations using a modified inviscid flux discretization," *International Journal for Numerical Methods in Fluids*, Vol. 57, No. 3, 2008, pp. 265–293.

[7] Shakib, F., Hughes, T. J., and Johan, Z., "A new finite element formulation for computational fluid dynamics: X. The compressible Euler and Navier-Stokes equations," *Computer Methods in Applied Mechanics and Engineering*, Vol. 89, No. 1–3, 1991, pp. 141–219, Second World Congress on Computational Mechanics.

[8] Venkatakrishnan, V., Allmaras, S. R., Kamenetskii, D. S., and Johnson, F. T., "Higher order schemes for the compressible Navier-Stokes equations," *16th AIAA Computational Fluid Dynamics Conference*, Orlando, FL, USA, June 2003, AIAA-2003-3987.

[9] Stroustrup, B., *The C++ Programming Language*, Addison-Wesley, Boston, MA, USA, 4th ed., 2013.

[10] *https://code.google.com/p/googletest/*.

[11] *http://valgrind.org/*.

[12] *https://gcc.gnu.org*.

[13] *http://ltp.sourceforge.net/coverage/lcov.php*.

[14] Spalart, P. and Allmaras, S., "A One-Equation Turbulence Model for Aerodynamic Flows," *Le Recherche Aerospatiale*, Vol. 1, 1994, pp. 5 – 21.

[15] Allmaras, S., Johnson, F., and Spalart, P., "Modifications and Clarifications for the Implementation of the Spalart-Allmaras Turbulence Model," *Proceedings of the 7th International Conference on Computational Fluid Dynamics, Big Island, HW*, 2012, ICCFD7-1902.

[16] Spalart, P., "Trends in Turbulence Treatments," *AIAA Fluids Denver, CO*, 2000, AIAA 2000-2306.

[17] Spalart, P., Deck, S., Shur, M., Squires, M., Strelets, M., and Travin, A., "A New Version of Detached-Eddy Simulation, Resistant to Ambiguous Grid Densities," *Theor. Comput. Fluid Dyn.*, Vol. 20, May 2006, pp. 181 – 195.

[18] *http://www.open-mpi.org*.

[19] Glasby, R., Burgess, N., Anderson, W., Wang, L., Mavriplis, D., and Allmaras, S., "Comparison of SU/PG and DG Finite-Element Techniques for the Compressible Navier-Stokes Equations on Anisotropic Unstructured Meshes," *AIAA: 51st Aerospace Sciences Meeting*, 2013, AIAA 2013-691.

[20] Burgess, N. and Glasby, R., "Advances in Numerical Methods for CREATE ™-AV Analysis Tools," *AIAA: 52nd Aerospace Sciences Meeting*, 2014, AIAA 2014-0417.

[21] Erwin, J., Anderson, W., Wang, L., and Kapadia, S., "High-Order Finite-Element Method for Three-Dimensional Turbulent Navier-Stokes," AIAA 2013-2571.

[22] Erwin, J., Anderson, W., Kapadia, S., and Wang, L., "Three-Dimensional Stabilized Finite Elements for Compressible Navier-Stokes," *AIAA Journal*, Vol. 51, 2013, pp. 1404.

[23] Hughes, T. J. R., Franca, L. P., and Hulbert, G. M., "A new finite element formulation for computational fluid dynamics: VIII. The Galerkin/least-squares method for advective-diffusive equations," *Computer Methods in Applied Mechanics and Engineering*, Vol. 73, No. 2, 1989, pp. 173–189.

[24] Hughes, T., Feijoo, G., Mazzei, L., and Quincy, J.-B., "The Variational Multiscale Method - A Paradigm for Computational Mechanics," *Computer Methods in Applied Mechanics and Engineering*, Vol. 166, 1998, pp. 3 – 24.

[25] Wheeler, M., "An Elliptic Collocation-Finite Element Method with Interior Penalties," *SIAM Journal of Numerical Analysis*, Vol. 15, 1978, pp. 152 – 161.

[26] Devine, K., Boman, E., Riesen, L., Catalyurek, U., and Chevalier, C., "Getting Started with Zoltan: A Short Tutorial," *Proc. of 2009 Dagstuhl Seminar on Combinatorial Scientific Computing*, 2009, Also available as Sandia National Labs Tech Report SAND2009-0578C.

[27] Saad, Y. and Schultz, M. H., "GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems," *SIAM Journal on Scientific and Statistical Computing*, Vol. 7, No. 3, 1986, pp. 856–869.

[28] Stoer, J. and Bulirsch, R., *Introduction to Numerical Analysis*, Springer-Verlag, New York, NY, USA, 3rd ed., 2002.

[29] Philip, B. and Chartier, T., "Adaptive Algebraic Smoothers," *Journal of Computational and Applied Mathematics*, Vol. 236, 2012, pp. 2277 – 2297.

[30] Galbraith, M., Allmaras, S., and Darmofal, D., "A Verification Driven Process for Rapid Development of CFD Software," *53rd AIAA Aerospace Sciences Meeting*, 2015, AIAA 2015 - 0818.

[31] Ceze, M., *A Robust hp-Adaptation Method for Discontinuous Galerkin Discretization Applied to Aerodynamic Flows*, Ph.D. thesis, University of Michigan, 2013.