

A Chemically Reacting Flow Solver for Generalized Grids

Edward A. Luke*, Xiao-Ling Tong†, Junxiao Wu‡, Lin Tang§ and Pasquale Cinnella¶
 Mississippi State, MS 39762, USA

A software system that supports the development of flexible multi-physics models is briefly described. This system is used in the creation of a finite volume method for three-dimensional generalized grids (unstructured grids composed of arbitrary polyhedra), targeting non-equilibrium flow scenarios, including complex thermodynamic phenomena and chemical reactions. Multi-component transport models for viscosity, conductivity, and molecular diffusion are briefly described, and turbulent mixing is modeled using the Baldwin-Lomax, Spalart-Allmaras, and Shear-Stress Transport (SST) approaches. Upwind flux differencing algorithms of the Roe family are employed to accommodate flow discontinuities such as shocks and slip lines. An implicit, second-order time integration scheme is utilized to achieve high efficiency for boundary layer and heat transfer simulations. Numerical techniques employed for solving problems related to poor matrix conditioning are also documented. Benchmark cases for turbulent flow over a transonic ONERA-M6 wing and hydrogen-oxygen combustion in an Rocket-Based Combined Cycle engine are used to illustrate the accuracy and robustness of the solution algorithm.

Nomenclature

\mathcal{A}	cell face area	Re	Reynolds number
c_p	specific heat coefficient at constant pressure	R_s	species gas constant
D_s	species diffusion coefficient	Sc	Schmidt number
e_0	total energy of mixture	t	time
$e_{internal}$	internal energy of mixture	T	temperature
e_s	species internal energy	\vec{u}	velocity vector
F_i	inviscid flux vector	\vec{u}_{Ω}	deformation velocity of a control surface
F_v	viscous flux vector	\mathcal{V}	cell volume
h_{f_s}	species heat of formation	\vec{V}_s	species mass diffusion velocity
h_s	species enthalpy, $h_s = e_s + R_s T$	\vec{W}	chemistry source term vector
\tilde{I}	identity tensor	\dot{w}_s	species chemical production rate
k	turbulent kinetic energy	x_i	Cartesian coordinates
$k_{b,r}$	backward reaction rate	X_s	species concentration, $X_s = \rho_s / \mathcal{M}_s$
$k_{f,r}$	forward reaction rate	y	normal distance to viscous wall
$K_{e,r}$	reaction equilibrium constant	Y_s	species mass fraction, $Y_s = \rho_s / \rho$
\mathcal{M}_s	species atomic mass	$\theta_{v,s}$	species characteristic vibrational temperature
n	time step	λ	coefficient of thermal conductivity
\vec{n}	unit vector normal to a surface	μ	coefficient of dynamic viscosity
NE	number of edges of a cell face	μ_t	coefficient of turbulent viscosity
NF	number of faces of a cell	ν	coefficient of kinematic viscosity, $\nu = \mu / \rho$
NR	number of chemical reactions	ν_t	coefficient of eddy viscosity, $\nu_t = \mu_t / \rho$
NS	number of chemical species	$\nu_{s,r}^i$	stoichiometric coefficient for reactants
NVT_s	number of vibrational modes for a species	$\nu_{s,r}$	stoichiometric coefficient for products
p	pressure	ρ	density of mixture
Pr	Prandtl number	ρ_s	species density
\vec{q}	heat conduction vector	$\tilde{\tau}$	stress tensor
q_p	vector of primitive variables	Ω	magnitude of the vorticity vector
Q	vector of conservative variables	Ω_c	control volume of cell c
R	numerical residual	$\partial\Omega_c$	boundary of cell c

*Assistant Professor, Department of Computer Science and Engineering

†Assistant Research Professor, SimCenter

‡Assistant Research Professor, Center for Advanced Vehicular Systems

§Senior Research Associate, SimCenter

¶Professor, Department of Aerospace Engineering

Introduction

Flows with chemical reactions are of interest across a wide range of engineering applications: examples include combustion, space vehicle reentry, and rocket plume problems. With the development of modern computers,

a great amount of effort has been focused in the areas of algorithm development and thermo-chemical modeling for these flows. A relatively recent review by Cinnella and Grossman¹ provides details on numerical methods for chemically reacting flows. There is still a continuing effort to better understand these complex flow phenomena.

The purpose of this study is to introduce a novel computational fluid dynamics (CFD) software framework, Loci,^{2,3} and apply it to the simulations of non-equilibrium flows. The Loci system uses a rule-based approach to automatically assemble the numerical simulation components into a working solver. This technique enhances the flexibility of simulation tools, reducing the complexity of CFD software induced by various boundary conditions, complex geometries, as well as varied physical models. Loci plays a central role in building flexible goal-adaptive algorithms that can quickly match numerical techniques with various physical modeling requirements. The results presented in this study provide the building blocks for the simulation of more complex flow problems, and also help to validate the viability of the Loci system.

In the following, a brief introduction of the Loci system is provided, and the development and validation of the reactive-flow application (CHEM) is detailed. The governing equations, including chemistry, thermodynamics, and turbulence models employed in the study are described next. The numerical formulation in three-dimensional generalized coordinates is then presented, including the treatment of inviscid and viscous fluxes, gradient construction, Jacobian formulations, and time integration. Whenever appropriate, numerical heuristics used in the study to overcome some numerical instabilities are also documented.

The Loci Framework

Loci is a framework for intra-application coordination of fine-grained numerical kernels and methods. To contrast, approaches such as MDICE⁴ and NPSS⁵ are focused more on inter-application coordination. Both levels of coordination (inter- and intra-application) are valuable with respect to software reuse. In the early to mid-Nineties there were many attempts to use object-oriented technology at the fine grain, with some success. For example, an early implementation of the ALEGRA⁶ code at Sandia National Labs employed a finite-element ALE approach (Arbitrary Lagrangian Eulerian), where objects represent fundamental numerical components such as tensors, material models, stresses, forces, etc. While the code featured an excellent object-oriented design, maximizing reuse by creating composable objects with simple semantics, the resulting performance was disappointing: this implementation of the ALEGRA code was more than ten times slower than traditional Fortran codes. The main sources of performance bottlenecks were in the use of operator overloading and dynamic dispatch. A later re-implementation of ALEGRA removed the use of operator overloading and reduced the use of dynamic dispatch. This implementation was able to achieve performances comparable to Fortran codes, at a

cost of significantly reducing the flexibility of the resulting design. More recently, techniques such as expression templates and tools such as PETE⁷ (also from Sandia) and Blitz++⁸ have been able to avoid the penalties associated with operator overloading. However, the costs of using dynamic dispatch at the lowest level of an application design continue to represent a fundamental optimization problem (due to the fact that it hides potential optimizations in register allocation and instruction reordering).

Loci allows one to have the flexibility of creating abstractions using fundamental composable objects with simple semantics, without inducing the costs associated with dynamic dispatch. It accomplishes this by introducing a run-time logical deduction engine that is capable of performing deductions on aggregates of simple types. The semantics of these aggregations are documented as a Loci *rule*, which is used to deduce loop bounds that are passed into computational subroutines. Using this technique, modern compilers are able to perform register scheduling, loop unrolling, and instruction scheduling to achieve performance. The deduction engine itself induces a very small overhead, since deductions on aggregates can be performed in $O(1)$ time in most cases. For example, in the CHEM code (to be introduced in the next section), the deduction overhead consumes significantly less than 1% of the overall execution time.

The advantages of this approach are numerous: 1) since the rules represent fundamental computational components their semantics are simple and easily captured; 2) since the semantics are simple, rules can be composed automatically using logical deduction; 3) the semantics of applications formed by these compositions can be (and are) automatically checked for internal consistency; and 4) intra-application resources management is possible, including automatic parallelization, cache optimization, memory management, and check-pointing.

The CHEM code

The CHEM code is a library of Loci rules (fine-grained components), and provides: primitives for generalized grids, including metrics; operators such as gradient; chemically reacting physics models such as equations of state, inviscid flux functions, and transport functions (viscosity, conduction, and diffusion); a variety of time and space integration methods; linear system solvers; and more. Combined with this library of rules is an application front end that generates an initial fact database and query required for Loci to assemble these rules into an application that can simulate three-dimensional flows of chemically reacting mixtures of thermally perfect gases. Moreover, it is more than an application that can simulate chemically reacting flows: it is a library of reusable components that can be dynamically reconfigured to solve a variety of problems involving generalized grids by changing the given fact database, adding rules, or changing the query.

For example, suppose that one wished to couple an application that provided a temperature field to an application

that required heat fluxes. One could provide the temperature field, grid, and conduction function in the fact database and query for `heat_flux` and the code would become a heat flux calculator by extracting those components of the chemically reacting flow code that compute heat fluxes. If the fact database provided inconsistent information (e.g., heat fluxes were impossible to compute with the given facts, or no unique solution was found), then the system would automatically generate error messages warning the user of an incomplete or inconsistent formulation. The likely components that would be used would include grid metrics, temperature gradients, conduction coefficients, etc. The user would not need to be concerned with these details. Loci would automatically extract the appropriate components and assemble them in the required manner as prescribed by the interface specifications given by each rule. The above represents just one of a myriad of possible examples.

Multi-Physics Simulations using Loci

Developments in multi-physics or multi-disciplinary simulations can be divided into two general approaches: loose coupling of a variety of applications that are specialized to single disciplines, hopefully iterating to convergence; or tight coupling in one integrated multidisciplinary application. The former approach has the advantage that the coupled code employs appropriate numerical models that have been validated for each individual discipline, and is relatively straightforward to assemble. However, it can have uncertain stability properties, and may be problematic for transient problems, particularly when characteristic time scales of the various disciplines are similar. The latter approach typically places all disciplines under one numerical method umbrella (e.g. finite-element or finite-volume). Current examples include SPECTRUM (now part of ANSYS) and PHYSICA.⁹ This approach has the advantages that coupling is seamless, easily incorporating transient and non-linear solvers. However, one may have conditioning problems if disciplines have widely different time-scales, unless care is taken in formulation. Also, this one-size-fits-all approach removes the possibility of using the most appropriate numerical method for each individual discipline.

With Loci, a third approach is possible. Each discipline can use the numerical method that is best suited to its accurate simulation (as in the loosely coupled approach), while the interface between disciplines can take advantage of knowledge of the specific numerical methods (e.g., space and time integration) to develop a coupling that remains true to the physics and numerics. A full *range* of possible interface treatments can be implemented: from loose coupling techniques, to domain-decomposition methods, all the way to tight non-linear coupling.

However, not all applications have to be deconstructed into Loci rules in order to be reused within the Loci framework. Applications have their value, which is rather significant, in the fact that they have been validated for solving

specific classes of problems: any major restructuring would destroy this value. In such cases, Loci provides facilities to package applications as Loci rules. However, with such an approach one must be willing to accept certain compromises: the resulting software will not be able to take full advantage of the automatic resource management schemes available in Loci, and the full range of coupling and extensibility options will not be available. In this sense, Loci is meant to be complementary to application-level toolkits such as NPSS and MDICE. Loci applications can be easily transformed into new applications by providing new facts, rules, or queries. As such, in the context of systems such as NPSS and MDICE, Loci provides flexible adapter applications that are created on the fly by re-composing existing computational kernels already stored in a pre-defined but extensible rule database (for example rules for curl, divergence, gradient, equations of state, finite-element or finite-volume integration methods). In summary, Loci provides an interesting new approach to multi-physics simulations: it allows the user to choose an optimal point within the entire range of coupling strategies, anywhere from the tightly coupled single discretization approach to the loosely coupled multi-code solution.

Model Equations

At the present time, a non-equilibrium flow model is implemented in Loci. This model has been benchmarked for inviscid reactive flow;² results from viscous, turbulent problems are presented in this study. The governing equations and physical models for the fluid phase are introduced in the following sections. Algorithms for the simulation of the thermo-mechanical response of a solid phase are under development (a preliminary version of a finite-element model developed for this purpose has already been implemented within Loci¹⁰).

Governing Equations

A finite-volume procedure is applied to discretize the flow equations. After integration over a computational cell, the governing equations for a three-dimensional flow with non-equilibrium chemistry and equilibrium internal energy, written in vector form for an arbitrary control volume Ω_c (closed by a boundary $\partial\Omega_c$) are:

$$\frac{d}{dt} \int_{\Omega_c(t)} Q dV + \int_{\partial\Omega_c(t)} (F_i - F_v) dS = \int_{\Omega_c(t)} \dot{W} dV, \quad (1)$$

where the vectors of conservative state variables, Q , inviscid flux, F_i , viscous flux, F_v , and chemistry source term, \dot{W} , are given by:

$$Q = \begin{bmatrix} \rho_1 \\ \vdots \\ \rho_s \\ \vdots \\ \rho_{\text{NS}} \\ \rho \tilde{u} \\ \rho e_0 \end{bmatrix}, \quad F_i = \begin{bmatrix} \rho_1 \tilde{u} \cdot \tilde{n} \\ \vdots \\ \rho_s \tilde{u} \cdot \tilde{n} \\ \vdots \\ \rho_{\text{NS}} \tilde{u} \cdot \tilde{n} \\ (\rho \tilde{u} \tilde{u} + p \tilde{I}) \cdot \tilde{n} \\ (\rho e_0 + p) \tilde{u} \cdot \tilde{n} \end{bmatrix}, \quad (2)$$

$$F_v = \begin{bmatrix} -\rho_1 \tilde{V}_1 \cdot \tilde{n} \\ \vdots \\ -\rho_s \tilde{V}_s \cdot \tilde{n} \\ \vdots \\ -\rho_{\text{NS}} \tilde{V}_{\text{NS}} \cdot \tilde{n} \\ \tilde{\tau} \cdot \tilde{n} \\ (\tilde{u} \cdot \tilde{\tau} - \tilde{q} - \sum \rho_s h_s \tilde{V}_s) \cdot \tilde{n} \end{bmatrix}, \quad \dot{W} = \begin{bmatrix} \dot{w}_1 \\ \vdots \\ \dot{w}_s \\ \vdots \\ \dot{w}_{\text{NS}} \\ 0 \\ 0 \end{bmatrix}. \quad (3)$$

Thermodynamic Model

The pressure term p is undefined in the governing equations. In the present study, pressure is determined from Dalton's law, which states that the pressure of a mixture of gases is the sum of partial pressures of each individual species, and each species behaves as a thermally perfect gas. Thus the mixture pressure can be modeled as:

$$p = \sum_{s=1}^{\text{NS}} \rho_s R_s T. \quad (4)$$

In Eq. (4), pressure is related to gas temperature, which is determined from the internal energy, e_{internal} . Moreover,

$$e_0 = \frac{1}{2} \tilde{u} \cdot \tilde{u} + e_{\text{internal}} + e_{\text{other}}, \quad (5)$$

where the total energy e_0 appears in the vector of conserved variables, and e_{other} denotes additional energy terms possibly involved, such as turbulent kinetic energy if turbulence mixing is considered, or latent heat if there is phase change in the flow. The internal energy of a mixture is computed as the sum of internal energy of species, e_s :

$$e_{\text{internal}} = \sum_{s=1}^{\text{NS}} Y_s e_s. \quad (6)$$

If the internal energy of each species is assumed to be in thermodynamic equilibrium, and vibrational contributions are included in e_s by means of a simple harmonic oscillator formula,¹¹ then the species internal energy is represented as:

$$e_s = n_s R_s T + \sum_{v=1}^{\text{NVT}_s} \left(\frac{R_s \theta_{v,s}}{e^{\theta_{v,s}/T} - 1} \right) + h_{f,s}. \quad (7)$$

The translational and rotational contributions to the internal energy are included by using appropriate values for the constant n_s . Alternatively, the user can select piecewise 4th degree polynomial functions for species specific heat c_{p_s} represented as:

$$c_{p_s} = A_s + B_s T + C_s T^2 + D_s T^3 + E_s T^4, \quad (8)$$

where A_s, B_s, C_s, D_s , and E_s are empirically determined coefficients that can be obtained from standard references such as JANAF tables.¹² Similarly, the Shomate curve fit can be used for species energy e_s :

$$c_{p_s} = A_s + B_s T + C_s T^2 + D_s T^3 + G_s T^{-2}, \quad (9)$$

where A_s, B_s, C_s, D_s , and G_s are empirically determined coefficients that can be obtained from standard references such as the NIST Chemistry WebBook.¹³

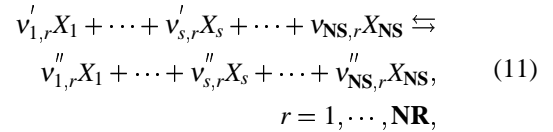
Species internal energy is obtained from these curve fits by integrating c_p using the expression

$$e_s = \int c_{p_s}(T) dT - R_s T, \quad (10)$$

where the constant of integration is provided by a user specified enthalpy at a reference temperature.

Chemistry Model

In the governing equations, the species production rates \dot{w}_s have to be modeled. In general, **NR** chemical reactions involving **NS** species can be represented as:



where X_s represents the species s . The species production rate can be expressed as:

$$\begin{aligned} \dot{w}_s &= \left(\frac{d\rho_s}{dt} \right)_{\text{chemistry}} = \mathcal{M}_s \sum_{r=1}^{\text{NR}} (v''_{s,r} - v'_{s,r}) \times \\ &\quad \left[k_{f,r} \prod_{l=1}^{\text{NS}} \left(\frac{\rho_l}{\mathcal{M}_l} \right)^{v'_{l,r}} - k_{b,r} \prod_{l=1}^{\text{NS}} \left(\frac{\rho_l}{\mathcal{M}_l} \right)^{v''_{l,r}} \right]. \end{aligned} \quad (12)$$

The forward reaction rates are evaluated by Arrhenius curve fits:

$$k_{f,r}(T) = CT^\eta e^{-\theta/T}, \quad (13)$$

where C , η , and θ are appropriate constants, and the backward reaction rates are obtained by using the following relationship:

$$k_{b,r} = \frac{k_{f,r}}{K_{e,r}}, \quad (14)$$

where $K_{e,r}$ is the equilibrium constant, which is determined from thermodynamics.²

Modeling of Viscous Terms

In order to close the system of equations, the stress tensor, the heat flux vector, and the species diffusion velocities that appear in the viscous flux must be defined. Only Newtonian fluids are considered here, where there is a linear relationship between stress and deformation rate. Moreover, Fourier's Law is employed to relate heat conduction and temperature gradients. Under these assumption, the

stress tensor and heat flux vector in Cartesian form can be written as:

$$\tau_{ij} = (\mu + \mu_t) \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) - \frac{2}{3} [(\mu + \mu_t) \nabla \cdot \tilde{u}] \delta_{ij}, \quad (15)$$

$$i = 1, 2, 3, \quad j = 1, 2, 3,$$

$$\tilde{q} = (\lambda + \mu_t c_p / Pr_t) \nabla T, \quad (16)$$

where δ_{ij} is the Kronecker tensor. A constant turbulent Prandtl number of $Pr_t = 0.9$ will be used in all computations.

The transport properties of the mixture, namely viscosity coefficient, μ , and thermal conductivity, λ , are usually evaluated in two steps: first, one determines the transport properties for each species; then a mixing rule is invoked in order to obtain mixture values.

Two models are applied to compute species transport properties. At temperature lower than 1000 K, Sutherland's law is used as:

$$t_i = T^{3/2} \frac{F_{t,i}}{T + G_{t,i}}, \quad (17)$$

where t_i stands for either μ_i or λ_i , and $F_{t,i}$, $G_{t,i}$ are constants determined empirically. At temperature higher than 1000 K, a more accurate model based on curve fit tabulation proposed by Gupta *et al.*¹⁴ is utilized:

$$\mu_i = \exp(C_{\mu,i}) T^{A_{\mu,i} \ln T + B_{\mu,i}}, \quad (18)$$

$$\lambda_i = \exp(E_{f,i}) T^{(A_{f,i} (\ln T)^3 + B_{f,i} (\ln T)^2 + C_{f,i} \ln T + D_{f,i})}, \quad (19)$$

where $A_{\mu,i}$, $B_{\mu,i}$, $C_{\mu,i}$, $A_{f,i}$, $B_{f,i}$, $C_{f,i}$, $D_{f,i}$ and $E_{f,i}$ are tabulated curve fit coefficients. Alternatively, 4th degree polynomial curve fits similar to Eq. (8) can be specified in place of Eqs. (18) and (19), and the coefficients of these curve fits can be obtained using the CHEMKIN transport library.¹⁵

Once the transport properties for individual species are obtained, Wilke's rule is applied to determine mixture values, as follows:

$$t = \sum_{i=1}^{NS} W_i t_i, \quad (20)$$

where t denotes transport properties for the mixture (either μ or λ) and the weighting function W_i is given by

$$W_i = \frac{X_i}{\sum_{j=1}^{NS} X_j \phi_{ij}}, \quad (21)$$

where the coefficient ϕ_{ij} is given by

$$\phi_{ij} = \frac{1}{\sqrt{8}} \left(1 + \frac{\mathcal{M}_i}{\mathcal{M}_j} \right)^{-1/2} \left[1 + \sqrt{\frac{\mu_i}{\mu_j}} \left(\frac{\mathcal{M}_i}{\mathcal{M}_j} \right)^{1/4} \right]^2. \quad (22)$$

Fick's law of diffusion is employed to model species diffusion velocity \tilde{V}_s :

$$\rho_s \tilde{V}_s = - \left(\rho D_s + \frac{\mu_t}{Sc_t} \right) \nabla Y_s, \quad (23)$$

where D_s is the diffusion coefficient, μ_t is the eddy viscosity, and Sc_t is the turbulent Schmidt number. The species diffusion coefficient, D_s , can be obtained by user specification or the CHEMKIN transport library.¹⁵

Turbulence Models

Several turbulence models are implemented in CHEM, including the algebraic Baldwin-Lomax,¹⁶ the one-equation Spalart-Allmaras,¹⁷ and a family of two-equation models including the SST formulation by Menter.¹⁸ These models are briefly described in the following sections.

Baldwin-Lomax Model

The Baldwin-Lomax model is a two-layer algebraic eddy viscosity model. The eddy viscosity is defined as

$$\nu_t = \begin{cases} \nu_{ti}, & y \leq y_{crossover}; \\ \nu_{to}, & y > y_{crossover}. \end{cases} \quad (24)$$

where y is the normal distance from the wall and $y_{crossover}$ is the smallest value of y for which $\nu_{ti} = \nu_{to}$. The inner and outer layer viscosities are given as follows:

Inner Layer:

$$\nu_{ti} = l^2 \Omega, \quad (25)$$

where the mixing length is

$$l = ky [1 - \exp(-y^+ / A^+)], \quad (26)$$

and Ω is the magnitude of the vorticity vector. The dimensionless normal distance is defined as

$$y^+ = \frac{u_\tau y}{\nu}, \quad (27)$$

where u_τ is the friction velocity ($u_\tau = \sqrt{\tau_{wall} / \rho}$).

Outer Layer:

$$\nu_{to} = K C_{cp} F_{wake} F_{kleb}(y), \quad (28)$$

where

$$F_{wake} = \min(y_{max} F_{max}, C_{wk} y_{max} u_{dif}^2 / F_{max}), \quad (29)$$

$$F_{max} = \frac{1}{k} \max(l \Omega), \quad (30)$$

and y_{max} is the value of y at which F_{max} occurs. The function $F_{kleb}(y)$ is the Klebanoff intermittency factor, given by

$$F_{kleb}(y) = \left[1 + 5.5 \left(\frac{C_{kleb} y}{y_{max}} \right)^6 \right]^{-1}. \quad (31)$$

The quantity u_{dif} in Eq. (29) is the difference between maximum and minimum velocity in the boundary layer velocity profile.

The constants appearing in the above relations are:

$$\begin{aligned} A^+ &= 26, & C_{cp} &= 1.6, & C_{kleb} &= 0.3, \\ C_{wk} &= 0.25, & k &= 0.4, & K &= 0.0168. \end{aligned} \quad (32)$$

At the implementation level, the Baldwin-Lomax model usually relies on having velocity and vorticity profiles on a smooth grid line, roughly orthogonal to the no-slip surface. Thus, due to this *non-local* feature of the model, there is a challenge if it is implemented with unstructured grids. However, by creating a mapping between cells and the nearest no-slip faces under the Loci system, an arbitrary grid line can be drawn, connecting a specific no-slip surface and all the cells which are related to this face.

Spalart-Allmaras Model

The defining equations for this model are written as follows:

$$\text{Kinematic Eddy Viscosity: } \bar{v} = v_t / f_{v1}.$$

$$\text{Eddy Viscosity Equation:}$$

$$\begin{aligned} \frac{\partial \bar{v}}{\partial t} + u_j \frac{\partial \bar{v}}{\partial x_j} - \frac{1}{\sigma} \frac{\partial}{\partial x_k} \left[(v + \bar{v}) \frac{\partial \bar{v}}{\partial x_k} \right] - \frac{c_{b2}}{\sigma} \frac{\partial \bar{v}}{\partial x_k} \frac{\partial \bar{v}}{\partial x_k} \\ = c_{b1} \tilde{S} \bar{v} - c_{w1} f_w \left(\frac{\bar{v}}{y} \right)^2, \end{aligned} \quad (33)$$

where the last two terms on the left hand side represent turbulent diffusion, and tensor notation is employed (repeated indexes j, k denote summations). The first term on the right is the turbulence production, while the second term denotes the destruction due to the presence of a wall.

Closure Coefficients and Auxiliary Relations:

$$\begin{aligned} c_{b1} = 0.14, \quad c_{b2} = 0.6, \quad c_{v1} = 7.1, \quad \sigma = 2/3, \\ c_{w1} = \frac{c_{b1}}{\kappa^2} + \frac{(1 + c_{b2})}{\sigma}, \quad c_{w2} = 0.3, \quad c_{w3} = 2, \quad \kappa = 0.41, \\ f_{v1} = \frac{\chi^3}{\chi^3 + c_{v1}^3}, \quad f_{v2} = 1 - \frac{\chi}{1 + \chi f_{v1}}, \\ f_w = g \left[\frac{1 + c_{w3}^6}{g^6 + c_{w3}^6} \right]^{1/6}, \quad \chi = \frac{\bar{v}}{v}, \quad g = r + c_{w2}(r^6 - r), \\ r = \frac{\bar{v}}{\tilde{S} \kappa^2 y^2}, \quad \tilde{S} = S + \frac{\bar{v}}{\kappa^2 y^2} f_{v2}, \quad S = \sqrt{2 \Omega_{ij} \Omega_{ij}}. \end{aligned} \quad (34)$$

The tensor $\Omega_{ij} = \frac{1}{2}(\partial u_i / \partial x_j - \partial u_j / \partial x_i)$ is one half of the vorticity tensor, and y is the distance to the closest wall surface. For simplicity, no tripping term is included.

The value \bar{v} at the wall boundary is set to zero, and the value of v_t in the freestream is selected as $v_t = 10^{-3} v$.

The corresponding integral form of Eq. (33) can be included into the system of governing equations, Eq. (1), by adding the following additional terms to the vectors Q , F , F_v , and W :

$$\begin{aligned} Q = \rho \bar{v}, \quad F = \rho \bar{v} \tilde{u} \cdot \tilde{n}, \quad F_v = \frac{1}{\sigma} \rho (v + \bar{v}) \nabla \bar{v} \cdot \tilde{n}, \\ \dot{W} = \rho c_{b1} \tilde{S} \bar{v} - \rho c_{w1} f_w \left(\frac{\bar{v}}{y} \right)^2 \\ + \frac{\rho}{\sigma} \left[(v + \bar{v}) \nabla \bar{v} \cdot \nabla \rho - c_{b2} \rho (\nabla \bar{v})^2 \right], \end{aligned} \quad (35)$$

where turbulent production, destruction, and part of diffusion are included in the source term.

Baseline Model (BSL)

It is well known that two-equation eddy-viscosity *low-Reynolds-number* turbulence models are among the most widely used models for engineering applications today, and the $k - \varepsilon$ model with damping functions near the wall is the most popular. However, the $k - \varepsilon$ model often suffers from numerical stability problems due to disparate turbulent time scales. Another well-known two-equation turbulence model is the $k - \omega$ model, developed by Wilcox.¹⁹

It has the advantage that it does not require damping functions in the viscous sublayer and that the equations are less stiff near the wall, therefore it is superior to the $k - \varepsilon$ model with regard to numerical stability. However, when applied to the free shear layers, it is found that there is a strong dependency of the results on the freestream value of ω .^{18,20} Menter created a new model, called baseline (BSL) model, by blending the $k - \varepsilon$ and $k - \omega$ models.²¹ It utilizes the $k - \omega$ model in the wall region and gradually switches to the $k - \varepsilon$ model away from the wall. To achieve this, the $k - \varepsilon$ model is first transformed into a $k - \omega$ formulation, and an additional cross diffusion term is added (another diffusion term associated with turbulent kinetic energy is neglected in the formulation under certain assumptions²²). The original $k - \omega$ equations are then multiplied by a blending function F_{bsl} , the transformed $k - \varepsilon$ equations are multiplied by $(1 - F_{bsl})$, and then both are added together. The blending function F_{bsl} is designed so that it is unity at the wall, and gradually approaches zero away from the wall. Note that the $k - \omega$ model can be easily obtained by setting $F_{bsl} = 1$ identically. In order to accurately predict adverse pressure gradient flows, especially in the wake region, Menter²¹ modified the BSL model by including the transport of the principal turbulent shear stress²³ in the eddy-viscosity formulations. This leads to the shear-stress transport (SST) model. In the present study, both BSL and SST models are discussed.

The defining equations for the BSL model are written as:

Kinematic Eddy Viscosity:

$$v_t = k / \omega, \quad (36)$$

Turbulent Stress Tensor:

$$\begin{aligned} \tau'_{ij} = \mu_t \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) - \frac{2}{3} (\mu_t \nabla \cdot \tilde{u} + \rho k) \delta_{ij}, \\ i = 1, 2, 3, \quad j = 1, 2, 3, \end{aligned} \quad (37)$$

Turbulent Kinetic Energy Equation:

$$\frac{D\rho k}{Dt} = \tau'_{ij} \frac{\partial u_i}{\partial x_j} - \beta^* \rho \omega k + \frac{\partial}{\partial x_j} \left[(\mu + \mu_t \sigma_k) \frac{\partial k}{\partial x_j} \right], \quad (38)$$

Turbulent Dissipation Equation:

$$\begin{aligned} \frac{D\rho \omega}{Dt} = \frac{\gamma}{v_t} \tau'_{ij} \frac{\partial u_i}{\partial x_j} - \beta \rho \omega^2 + \frac{\partial}{\partial x_j} \left[(\mu + \mu_t \sigma_\omega) \frac{\partial \omega}{\partial x_j} \right] \\ + 2(1 - F_{bsl}) \rho \sigma_{\omega 2} \frac{1}{\omega} \frac{\partial k}{\partial x_j} \frac{\partial \omega}{\partial x_j}. \end{aligned} \quad (39)$$

Closure Coefficients:

All the constants ϕ of the model are computed by blending the appropriate $k - \omega$ and $k - \varepsilon$ constants, as follows:

$$\phi = F_{bsl} \phi_1 + (1 - F_{bsl}) \phi_2, \quad (40)$$

where the constants $\phi_1 (k - \omega)$ are:

$$\begin{aligned} \sigma_{k1} &= 0.5, & \sigma_{\omega1} &= 0.5, & \beta_1 &= 0.075, \\ \beta^* &= 0.09, & \kappa &= 0.41, & \gamma_1 &= \beta_1/\beta^* - \sigma_{\omega1}\kappa^2/\sqrt{\beta^*}, \end{aligned} \quad (41)$$

and the constants $\phi_2 (k - \varepsilon)$ are:

$$\begin{aligned} \sigma_{k2} &= 0.5, & \sigma_{\omega2} &= 0.856, & \beta_2 &= 0.0828, \\ \beta^* &= 0.09, & \kappa &= 0.41, & \gamma_2 &= \beta_2/\beta^* - \sigma_{\omega2}\kappa^2/\sqrt{\beta^*}. \end{aligned} \quad (42)$$

The blending function F_{bsl} is defined as follows:

$$F_{bsl} = \tanh(\arg_{bsl}^4), \quad (43)$$

where

$$\arg_{bsl} = \min \left[\max \left(\frac{\sqrt{k}}{0.09\omega y}, \frac{500\nu}{y^2\omega} \right), \frac{4\rho\sigma_{\omega2}k}{CD_{k\omega}y^2} \right], \quad (44)$$

and y is the distance to the closest point away from the wall surface. In the above, $CD_{k\omega}$ is defined as:

$$CD_{k\omega} = \max \left(2\rho\sigma_{\omega2} \frac{1}{\omega} \frac{\partial k}{\partial x_j} \frac{\partial \omega}{\partial x_j}, 10^{-20} \right). \quad (45)$$

The boundary conditions for k and ω at a solid wall are:

$$k = 0, \quad \omega = 10 \frac{6\gamma}{\beta_1(\Delta y_1)^2}, \quad (46)$$

where Δy_1 is the distance from the first cell center to the solid wall.

The following freestream values are used in the current simulations:

$$\omega_\infty = 10 \frac{U_\infty}{L}, \quad \nu_{t\infty} = 10^{-3} \nu_\infty, \quad (47)$$

where U_∞ is the reference velocity, ν_∞ is the laminar viscosity at reference conditions, and L is the geometry reference length.

The corresponding integral form of Eqs. (38) and (39) can be included into the system of governing equations, Eq. (1), by adding the following additional terms to the vectors Q , F , F_v , and W :

$$\begin{aligned} Q &= \begin{bmatrix} \rho k \\ \rho \omega \end{bmatrix}, \quad F = \begin{bmatrix} \rho k \tilde{u} \cdot \tilde{n} \\ \rho \omega \tilde{u} \cdot \tilde{n} \end{bmatrix}, \quad F_v = \begin{bmatrix} (\mu + \mu_t \sigma_k) \nabla k \cdot \tilde{n} \\ (\mu + \mu_t \sigma_\omega) \nabla \omega \cdot \tilde{n} \end{bmatrix}, \\ W &= \rho \begin{bmatrix} \tau'_{ij} \frac{\partial u_i}{\partial x_j} - \beta^* \rho \omega k \\ \frac{\gamma}{\nu_t} \tau'_{ij} \frac{\partial u_i}{\partial x_j} - \beta \rho \omega^2 + 2(1 - F_{bsl}) \rho \sigma_{\omega2} \frac{1}{\omega} \frac{\partial k}{\partial x_j} \frac{\partial \omega}{\partial x_j} \end{bmatrix}. \end{aligned} \quad (48)$$

Shear Stress Transport Model (SST)

The SST model is similar to the BSL model described above, except that $\sigma_{k1} = 0.85$, and the eddy viscosity is defined as:

$$\nu_t = \frac{a_1 k}{\max(a_1 \omega, \Omega F_{sst})}, \quad (49)$$

where Ω is the absolute value of the vorticity, and $a_1 = 0.31$. The blending function F_{sst} is given by:

$$F_{sst} = \tanh(\arg_{sst}^2), \quad (50)$$

where

$$\arg_{sst} = \max \left(2 \frac{\sqrt{k}}{0.09\omega y}, \frac{500\nu}{y^2\omega} \right). \quad (51)$$

Turbulence Compressibility Corrections

Compressibility corrections for high-speed shear flows^{24,25} are implemented in the turbulence equations. These corrections are expected to improve the results on some problems, such as the mixing of different chemical species in high-speed shear flows. Moreover, a clipping technique is applied to k and ω to ensure the positivity of the models. Additionally, a limiter on turbulent production is provided to keep the models sufficiently close to the equilibrium assumptions upon which they were derived. However, no further numerical modifications are made.

Numerical Formulation

The numerical model employed in this study is a finite-volume technique that supports generalized grids²⁶ (*generalized grids* are discretizations composed of arbitrary polyhedra, including tetrahedra, prisms, pyramids, and hexahedra). Since this formulation is based on cell centered integrations, any grid type can be expressed, including ‘‘hanging’’ nodes found in adaptive mesh refinement (AMR).

Numerical Approximations to Spatial Integrals

The numerical solution of the governing equations, Eq. (1), is obtained by applying the finite volume method. This approach is frequently used because it can guarantee that numerical truncation errors do not violate conservation properties. The numerical integration of Eq. (1) begins with approximations to volume and surface integrals. For the volume integrals a second-order midpoint rule is used. For example, the numerical integration of Q results in

$$\int_{\Omega_c(t)} Q(\tilde{x}, t) dV = Q_c(t) \mathcal{V}_c(t), \quad (52)$$

where $Q_c(t)$ is the value of Q at the centroid of cell c , and $\mathcal{V}_c(t)$ is defined by

$$\mathcal{V}_c(t) = \int_{\Omega_c(t)} dV. \quad (53)$$

The numerical integration of the surface integral in Eq. (1) is accomplished by summing the contributions of each of the **NF** faces of cell c . Each individual contribution

is again approximated using the midpoint rule. The flux function itself will require additional numerical treatment, and will be discussed in later sections. For now, assume that the flux can be approximated by a function, \hat{F} , of conservative values to the left and right of the face. Given this, the numerical integration of $F = F_i - F_v$ results in the following

$$\int_{\partial\Omega_c(t)} F dS = \sum_{f=1}^{NF_c} \int_{\partial\Omega_{c,f}(t)} F dS \approx \sum_{f=1}^{NF_c} \mathcal{A}_{c,f}(t) \hat{F}_f, \quad (54)$$

where the area of the face, $\mathcal{A}_{c,f}(t)$, is defined as

$$\mathcal{A}_{c,f}(t) = \int_{\partial\Omega_{c,f}(t)} dS. \quad (55)$$

At this point, Eq. (1) is numerically approximated by the equation

$$\frac{d}{dt} [\mathcal{V}_c(t) Q_c(t)] + \sum_{f=1}^{NF_c} \mathcal{A}_{c,f}(t) \hat{F}_f = \mathcal{V}_c(t) \dot{W}_c(t). \quad (56)$$

Notice that the differential term that remains in this equation applies to the product of volume and conservative state vector for the cell. However, the variable that is the objective of these calculations is $Q_c(t)$, not $\mathcal{V}_c(t) Q_c(t)$. This problem is solved by applying the chain rule:

$$\frac{d}{dt} Q_c(t) \mathcal{V}_c(t) = Q_c(t) \frac{d}{dt} \mathcal{V}_c(t) + \mathcal{V}_c(t) \frac{d}{dt} Q_c(t). \quad (57)$$

The derivative of volume with respect to time can be converted into a spatial integral through the use of an identity for integration over time-dependent domains:²⁷

$$\begin{aligned} Q_c \frac{d}{dt} \mathcal{V}_c(t) &= Q_c \frac{d}{dt} \int_{\Omega_c(t)} dV \\ &= Q_c \int_{\partial\Omega_c(t)} \tilde{u}_\Omega \cdot \tilde{n} dS \\ &\approx Q_c \sum_{f=1}^{NF_c} \mathcal{A}_{c,f}(t) (\tilde{u}_{\Omega,f} \cdot \tilde{n}_{c,f}). \end{aligned} \quad (58)$$

This equation, known as the geometric conservation law,²⁸ is necessary for correct time integration when mesh deformation is present. Given this, the solution method can now be described in terms of a system of ordinary differential equations of the form

$$\mathcal{V}_c \frac{d}{dt} Q_c = R_c, \quad (59)$$

where R_c is given by the expression

$$R_c = \mathcal{V}_c \dot{W}_c - \sum_{f=1}^{NF_c} \mathcal{A}_{c,f} \hat{F}_f - Q_c \sum_{f=1}^{NF_c} \mathcal{A}_{c,f} (\tilde{u}_{\Omega,f} \cdot \tilde{n}_{c,f}). \quad (60)$$

Eqs. (59) and (60) describe a system of ordinary differential equations that numerically model the time evolution

of the fluid dynamics equations when simultaneously satisfied for all cells in the mesh. To represent this fact, the cell subscript c will be dropped. Thus Q_c represents the fluid state for cell c , while Q represents the fluid states of all cells in the mesh. For example, while Eq. (59) represents the cell by cell differential equations, the global system of equations is given by

$$\mathcal{V} \frac{d}{dt} Q(t) = R(Q(t), t). \quad (61)$$

Geometric Integrations

In the previous section, numerical integrations over surfaces and volumes were described without explicitly defining the formulas used to evaluate Eqs. (53) and (55). Here the numerical integrations that approximate the volumes and areas of generalized cells and faces are discussed in some detail. Several considerations have to be made regarding these computations. First, when the nodes of a generalized face are not co-planar, the geometry of the face is not uniquely specified. A unique specification of the geometry of such faces can be obtained by subdividing the face into triangles: using a symmetric decomposition, each edge of the face, combined with the face centroid, creates a triangle. This strategy allows the face geometry to be independent of data-structures used to describe generalized meshes (the geometry is the same regardless of the ordering of the face edges). Using this technique, the area for the face is determined numerically using the following summation

$$\tilde{\mathcal{A}}_{c,f} = \frac{1}{2} \sum_{e=1}^{NE_f} (\tilde{x}_{1,e} - \tilde{x}_{c,f}) \times (\tilde{x}_{2,e} - \tilde{x}_{c,f}), \quad (62)$$

where $\tilde{x}_{1,e}$ and $\tilde{x}_{2,e}$ are the positions of the two nodes of edge e in counter-clockwise order, and $\tilde{x}_{c,f}$ is the face centroid (approximated by an edge-length-weighted sum of edge centers). Here the computed area is a vector, which is represented as the product of the face normal vector and the face area, $\tilde{\mathcal{A}} = \mathcal{A} \tilde{n}$.

For numerical approximations of the volume integrals, it is essential to achieve a geometric conservation of volume, avoiding truncation errors in integrating volumes that would yield inaccurate total volume computations. For example, one would expect that the sum of the individual cell volumes would be equal to the total grid volume if exact arithmetic were used. To accomplish this, the volume of a cell is determined by decomposing it into tetrahedra and summing the individual tetrahedral volumes. The volume of a tetrahedron is given by

$$\mathcal{V}_{tet} = \frac{1}{3!} (\tilde{a} \cdot (\tilde{b} \times \tilde{c})), \quad (63)$$

where \tilde{a} , \tilde{b} , and \tilde{c} are the three edge vectors from a given vertex of the tetrahedron. The volume of a generalized cell is then computed by using the triangulation of each face and the cell centroid: each surface triangle and the cell centroid define a tetrahedron. The cell volume is the sum of the

tetrahedra volumes. Thus, the volume of a cell is computed by combining Eqs. (62) and (63), and factoring, to obtain

$$\mathcal{V}_c = -\frac{1}{3} \sum_{f=1}^{\text{NF}_c} (\tilde{x}_{c,c} - \tilde{x}_{c,f}) \cdot \tilde{\mathcal{A}}_{c,f}, \quad (64)$$

where $\tilde{x}_{c,c}$ is the cell centroid (approximated by the area-weighted sum of face centroids). Notice that this computation assumes that the face normals point outward from the cell, which explains the negative sign. Using this approach, one can compute the correct volume of any cell, including highly non-convex cells.

Treatment of Inviscid Fluxes

The inviscid terms of the Navier-Stokes equations are treated using flux-difference-splitting techniques. The application of these approaches in the context of high-resolution finite-volume algorithms involves the reconstruction of functional values within each cell to provide *left* and *right* states at cell faces. When the function is relatively smooth, these left and right states are approximately equal. However, when a discontinuity occurs, the resulting fluxes should satisfy the Rankine-Hugoniot equations to remain consistent with the underlying physics. This is where the application of flux difference algorithms becomes salient. This section describes techniques used to establish left and right conditions for generalized grids.

Cell Function Reconstruction

A piecewise linear reconstruction is used to approximate the solution variables within cells. Primitive variables are used to ensure that these reconstructions remain physical (no negative temperatures or pressures). The linear reconstruction is derived from cell values and gradients by applying a second-order Taylor-series expansion:

$$q_p(\tilde{x}) = q_p(\tilde{x}_c) + \nabla q_p(\tilde{x}_c) \cdot \delta\tilde{r} + O((\delta\tilde{r})^2), \quad (65)$$

where ∇q_p is the gradient of the primitive variables q_p , and $\delta\tilde{r}$ is the vector from the center of the cell \tilde{x}_c to the desired point \tilde{x} .

Thus, a define piecewise linear reconstruction of the primitive variables in cell c with centroid $\tilde{x}_{c,c}$ is given by

$$q_{p,c}^R(\tilde{x}) = q_{p,c} + \nabla q_{p,c} \cdot \delta\tilde{r}, \quad (66)$$

where $q_{p,c}^R(\tilde{x})$ is the reconstructed function, $q_{p,c}$ is the computed primitive variable, and $\nabla q_{p,c}$ is the computed gradient within cell c .

The gradient at the cell centroid, $\nabla q_{p,c}$, is evaluated by minimizing the weighted error between the reconstructed function and neighboring cell values. Thus the error of the reconstructed function minimizes the weighted L_2 -error in the neighborhood of cell i :

$$\text{error} = \sqrt{\sum_{f=1}^{\text{NF}_c} (\mathcal{A}_{c,f} [q_{p,c}^R(\tilde{x}_{c,d}) - q_{p,d}]^2)}, \quad (67)$$

where $\mathcal{A}_{c,f}$ is the area of the common face shared by the adjoining cells c and d , and $\tilde{x}_{c,d}$ is the centroid of cell d . Area weighting of the error, inspired by a Green's theorem perspective of gradient computations, is used to recover correct thin-layer behavior for viscous grids applied to curved surfaces, where highly anisotropic prismatic cells are usually found. The gradient that minimizes Eq. (67) is obtained by using a standard linear least squares approach. A QR factorization method is used to solve the resulting (overdetermined) linear system for enhanced numerical stability.

However, the reconstruction given in Eq. (66) cannot be used for problems that contain discontinuities, due to the introduction of non-physical overshoots. The reconstruction is constrained to be monotonic for the inviscid flux extrapolations. This is accomplished by applying a gradient limiter to the reconstruction to form the limited reconstruction:

$$q_{p,c}^L(\tilde{x}) = q_{p,c} + \phi_c \nabla q_{p,c} \cdot \delta\tilde{r}, \quad (68)$$

where ϕ_c is the limiter function, that ideally has a value of unity when reconstructions are smooth but diminishes to zero in the presence of discontinuities. A variety of limiter functions can be employed: a few that are applicable to generalized grids are described below.

Barth and Jespersen Limiter

The limiter developed by Barth and Jespersen²⁹ was originally formulated, and is widely used, for multi-dimensional unstructured meshes. This limiter is easily extended to generalized grids. The limiter function is defined as follows:

$$\phi_c = \min(\phi_{cd}), \quad (69)$$

where

$$\phi_{cd} = \begin{cases} \min\left(1, \frac{q_p^{\max} - q_{p,c}}{q_p^R(\tilde{x}_d) - q_{p,c}}\right) & : q_{p,c} > q_p^R(\tilde{x}_{c,f}), \\ \min\left(1, \frac{q_p^{\min} - q_{p,c}}{q_p^R(\tilde{x}_d) - q_{p,c}}\right) & : q_{p,c} < q_p^R(\tilde{x}_{c,f}), \\ 1 & : q_{p,c} = q_p^R(\tilde{x}_{c,f}), \end{cases} \quad (70)$$

where q_p^{\min} and q_p^{\max} are the minimum and maximum values of q_p respectively, among the cells adjacent to cell c , including cell c itself, and $q_p^R(\tilde{x}_{c,f})$ is the cell reconstructed function, defined in Eq. (66), used to extrapolate from cell c to the face between cells c and d .

Venkatkrishna's Limiter

While the limiter of Barth and Jespersen prohibits overshoots, it often exhibits poor convergence characteristics, due to the appearance of limit cycles. In addition, it frequently destroys the accuracy of solutions in relatively smooth regions of the field, where small numerical perturbations activate the limiter. To correct these problems, Venkatkrishna³⁰ proposed a thresholded limiter. The thresholding in this limiter is designed so that it allows small overshoots in relatively smooth regions, while

strongly enforcing limiting where strong perturbations are present. The resulting limiter produces much better convergence rates with higher accuracy than the Barth limiter. The Venkatakrishtna limiter used in the CHEM code replaces Eq. (70) with the following:

$$\phi_{cd} = \frac{1}{\Delta_-} \left[\frac{(\Delta_+^2 + \varepsilon^2)\Delta_- + 2\Delta_-^2\Delta_+}{\Delta_+^2 + 2\Delta_-^2 + \Delta_+\Delta_- + \varepsilon^2} \right], \quad (71)$$

where

$$\varepsilon^2 = \left(\frac{Kl_c}{l_{ref}} \right)^3 q_{ref}^2, \quad (72)$$

and l_c is a reference length defined by cell c . Moreover, l_{ref} is a reference length defined by the total grid volume, and q_{ref} is a reference condition computed from the local cell density, sound-speed, or pressure. K is a user-defined parameter that sets a threshold for limiting. For this study, K is set to a value of 1.0. The other variables appearing in Eq. (71) are defined as follows:

$$\Delta_- = q_p^R(\tilde{x}_{c,f}) - q_{p,c}$$

and

$$\Delta_+ = \begin{cases} q_p^{max} - q_p^R(\tilde{x}_{c,f}) & : q_{p,c} < q_p^R(\tilde{x}_{c,f}) \\ q_p^{min} - q_p^R(\tilde{x}_{c,f}) & : q_{p,c} > q_p^R(\tilde{x}_{c,f}) \end{cases}. \quad (73)$$

An Adaptive Approximate Riemann Solver

The reconstruction method described earlier is used to establish conditions on either side of a given face in the mesh (the *left* and *right* states already mentioned). The numerical treatment of the inviscid terms takes advantage of the approximate Riemann solver algorithms to obtain low dissipation, shock-resolving numerical schemes. These algorithms involve solving approximations of the Riemann problem at cell faces. The most popular of these approximate Riemann solvers is the well known Roe scheme, which has been extended to chemically reacting flows.³¹ However, the Roe algorithm is not robust when dealing with slowly moving strong shocks. Unfortunately, these shocks occur frequently in problems involving chemistry: for example, bow shocks of a hypersonic vehicle or Mach diamonds in rocket exhausts. To resolve problems with strong shocks, an adaptive approach suggested by Quirk³² is employed: the Roe scheme is used in most regions, while a slightly more dissipative but more robust HLLE³³ algorithm is used in regions close to strong shocks. However, the original formulation of the adaptive scheme described by Quirk was set in the context of structured grids; an extension to generalized grids is proposed, as follows. First, strong shocks are identified by finding faces where significant pressure jumps exist, using the relation

$$\frac{|p_r - p_l|}{\min(p_l, p_r)} > \alpha, \quad (74)$$

where α is set to a value between 1 and 2, and the subscripts r and l refer to right and left states, respectively.

However, it is not sufficient to apply the HLLE scheme only at the strong shock: instead, it is necessary to apply it downstream of the shock. This is accomplished by first looping over cells, and marking the downstream cell of faces that satisfy Eq. (74). Then a loop over faces follows, marking cells that have adjacent upstream cells marked for strong shocks. The latter process is repeated a few times, and as a result a layer of a few cells downstream of the strong shock are marked (the number of repetitions is a user-defined parameter, and it typically ranges from 4 to 8). Finally, the HLLE scheme is employed for any face where the cell on either side has been marked as being close to a strong shock. This algorithm has the advantage that it can be applied to arbitrary grid types.

Treatment of Viscous Fluxes

The evaluation of the viscous fluxes requires careful consideration, since discontinuities are difficult to reconcile with computations of stresses and diffusion velocities. The considerations and concerns affecting the proper treatment of the viscous fluxes are different from those for the inviscid terms. In particular, the final integrated stencil (the sum of all of the numerical viscous fluxes for any given cell) must consist of positive coefficients if the numerical approximation of the diffusion process is to maintain the maximum principle associated with the Laplace equation. Essentially, these diffusion processes should not introduce new extrema in the solution. To ensure stencils with positive coefficients, the technique used in the *Cobalt*₆₀ code³⁴ is applied: it has the advantage of guaranteeing positive coefficients of the Laplace operator, at the cost of evaluating a non-zero result when applied to a linear function.

To evaluate the viscous fluxes, mixture density and velocity at each face are needed, as well as gradients of species mass fractions, velocities, and temperatures. Face values are evaluated by using a simple volume-weighted average of the integrated cell values on either side of the face. The evaluation of face gradients is more involved: the first step is the determination of the average of the least-square gradient (without limiter) computed for the cells on either side of the face, using the procedure already outlined for the inviscid flux evaluation. The second step is the computation of the gradient in the direction normal to the face. The final result is a combination of the previous two gradients:

$$\nabla\phi_f = \nabla\phi_{avg} - (\nabla\phi_{avg} \cdot \tilde{n})\tilde{n} + \frac{\phi(\tilde{x}_{c,c}) - \phi(\tilde{x}_{d,c})}{(\tilde{x}_{c,c} - \tilde{x}_{d,c}) \cdot \tilde{n}}\tilde{n}, \quad (75)$$

where ϕ is the quantity under consideration. In general, a simple averaging of cell gradients is insufficient, due to near zero coefficients in the Laplace stencil. The effect of this poor stencil is observed in high frequency oscillations in the solution that damp more slowly than what is physically meaningful. On the other hand, the cell centered differences do not have this problem, however they contain accurate gradient information in a single direction. The above blending provides both good stencils and true multidimensional gradients at the face.

The fact that this approach produces spurious source terms when applied to linear functions is somewhat disconcerting. However, it appears that this issue does not have a significant impact in practice.

Time Integration

The implicit time integration scheme employs a two-parameter family of algorithms, as described by Beam and Warming,³⁵ and is given by the equation

$$\begin{aligned} \mathcal{V}((1 + \psi)\Delta Q^n - \psi\Delta Q^{n-1}) = \\ \Delta t\{(1 - \theta)R^n(Q^n) + \theta R^{n+1}(Q^{n+1})\}, \end{aligned} \quad (76)$$

where n is the current time step and $\Delta Q^n = Q^{n+1} - Q^n$. In the above, θ and ψ are two parameters that determine the accuracy of the time-integration algorithms. For example, setting $\theta = 1, \psi = 0$ gives the implicit backward Euler scheme typically used in steady-state simulations, while a second-order three-point backward scheme ($\theta = 1, \psi = \frac{1}{2}$) is used for time-accurate simulations.

Eq. (76) represents a **non-linear** system of equations for the values of Q^{n+1} , and can be rearranged to read

$$\begin{aligned} \frac{1 + \psi}{\theta\Delta t}\mathcal{V}(Q^{n+1} - Q^n) - \\ \left[\frac{1 - \theta}{\theta}R^n(Q^n) + R^{n+1}(Q^{n+1}) \right] - \\ \mathcal{V}\frac{\psi}{\theta\Delta t}(Q^n - Q^{n-1}) = \mathcal{L}(Q^{n+1}) = 0. \end{aligned} \quad (77)$$

Eq. (77) is solved by a Newton iterative method, as follows:

$$\mathcal{L}'(Q^{n+1,p})(Q^{n+1,p+1} - Q^{n+1,p}) = -\mathcal{L}(Q^{n+1,p}), \quad (78)$$

for $p \geq 0$, where the Newton iteration is initialized using the previous time step value ($Q^{n+1,p=0} = Q^n$), and the Jacobian $\mathcal{L}'(Q^{n+1,p})$ is given by

$$\begin{aligned} \mathcal{L}'(Q^{n+1,p}) &= \frac{\mathcal{V}^{n+1}(1 + \psi)}{\theta\Delta t}I - \left[\frac{\partial}{\partial Q}R^{n+1,p}(Q) \right] \\ &= \frac{\mathcal{V}^{n+1}(1 + \psi)}{\theta\Delta t}I - \mathcal{V}^{n+1}\frac{\partial\dot{W}}{\partial Q} + \\ &\quad \sum_{f \in \text{faces}} \mathcal{A}_f^{n+1} \frac{\partial\hat{F}(Q_{l,f}, Q_{r,f})}{\partial Q} + \\ &\quad I \left[\sum_{f \in \text{faces}} \mathcal{A}_f^{n+1} (\tilde{u}_{\Omega,f} \cdot \tilde{n}_f) \right]. \end{aligned} \quad (79)$$

Eq. (78) is solved using a Gauss-Seidel iteration method.² The turbulent equations are decoupled from the mean flow solver, and communication of variables between turbulent and mean flow is established at the same time step. Compared with the coupled version (which was previously developed), the decoupled solver has actually improved numerical stability, since the coupling terms in the Jacobian matrices are more likely to yield ill-conditioned linear systems.

Local Time-Stepping Scheme

For solution of steady state problems, a local time-stepping scheme is used. A local time step for any given cell is chosen such that it is the smallest of the following three values: 1) a user-specified maximum value, 2) the value computed from a user-specified CFL condition, and 3) the time step required to produce no more than an estimated μ_{relax} percent change in temperature, pressure, or density.

Of the above three control variables, density is explicitly present in the vector of conserved variables, and pressure can be easily found from the same vector, plus the equation of state, Eq. (4), once the temperature is known. Consequently, a change in temperature during a given interval of time has to be estimated: a way to accomplish this is to solve an explicit time-integration step to describe a functional relationship between time and temperature. Specifically, an estimate for the value of Q after Δt time has elapsed is found by means of an explicit time integration step, as follows:

$$Q_{estimated}^{n+1} = Q(\Delta t) = Q^n + \Delta t R^n(Q^n). \quad (80)$$

Then, temperature is found from Q using Newton's method to solve Eqs. (5), (6), and (7) for T . The change in temperature for a given time step can be estimated by

$$\Delta T_{est} = T(Q_{estimated}^{n+1}) - T(Q^n). \quad (81)$$

If this estimate exceeds the maximum prescribed change, i.e. $|\Delta T_{est}| > \mu_{relax}T(Q^n)$, then the new time step is determined by solving

$$T(Q(\Delta t)) - T(Q^n)[1 + \text{sign}(\Delta T_{est})\mu_{relax}] = 0, \quad (82)$$

using Ridders' algorithm³⁶ to solve for Δt . Thus, the computation of the local time step requires two levels of iterative root solvers: Ridders' at the highest level, and Newton iterations to solve for temperatures. While this can be rather costly, it is only applied in regions that are changing too fast for the linearization to be appropriate. In these cases, it has been found that finding a suitable time step is well worth the extra computational effort, because it increases significantly the robustness of the overall algorithm.

Jacobian Formulations

The Jacobian matrix used in the Newton method, described by Eq. (79), consists of a block-diagonal matrix, formed from Jacobians of chemistry source terms, combined with both diagonal and off-diagonal matrices, formed from the differentiation of flux functions. When constructing this Jacobian, one takes advantage of the fact that the flux function, $\hat{F}(Q_l, Q_r)$, is defined by the conservative variables on the left and right side of the face. Then the Jacobian of this function will consist of two matrices, given

by

$$\begin{aligned} f_{jl} &= \mathcal{A}^{n+1} \frac{\partial \hat{F}(Q_l, Q_r)}{\partial Q_l}, \\ f_{jr} &= \mathcal{A}^{n+1} \frac{\partial \hat{F}(Q_l, Q_r)}{\partial Q_r}. \end{aligned} \quad (83)$$

The variables above, f_{jl} and f_{jr} , are Jacobians of the flux functions located at faces and are components of the overall Jacobian matrix given in Eq. (79). More details on the final form of the Jacobian matrix are given by Luke.²

Inviscid Flux Jacobians

In the study, all Jacobians are evaluated analytically, except for the inviscid flux, whereby Jacobians associated with the Roe scheme are too complex to implement efficiently. In this case, several alternatives are available: 1) use analytic Jacobians of simpler inviscid flux functions such as Steger-Warming or Van Leer, or 2) compute Jacobians of the Roe flux numerically. The analytic Van Leer Jacobian was found to provide superior stability properties, but severely hampered convergence of the implicit scheme. On the other hand, the numerical Roe Jacobians provided better convergence rates, but sometimes produced ill-conditioned linear systems. It was observed that these problems occurred in regions where discontinuities existed in the Roe Jacobians, due to the use of non-linear absolute value functions. A compromise that appears to provide a reasonable balance between convergence and robustness has been found by smoothing the Roe flux function before taking its derivative. This is accomplished by replacing the absolute value function $|x|$ with the continuous function $\sqrt{x^2 + \epsilon}$, where ϵ is set to .05 times the average sound speed.

Viscous Flux Jacobians

In the construction of viscous flux Jacobians, a thin-layer approximation is used in the representation of the face gradient by using simple centered differences in the direction of the vector connecting cell centroids on either side of the face.

Jacobians for Turbulence Models

In order to preserve the diagonal dominance of the matrix in the linear system, positive parts of the source terms are not linearized: only the negative parts (which model the destruction of turbulence) are included in the Jacobians. For the Jacobians used in the BSL, SST, and $k - \omega$ models the approach of Merci *et al.*³⁷ is adopted.

Results for Selected Problems

In the following, a few selected test cases are introduced in order to document the capabilities of the CHEM code. More extensive tests and validations have been conducted over the last several years, and some significant results have already been reported in the literature.^{2,38-43}

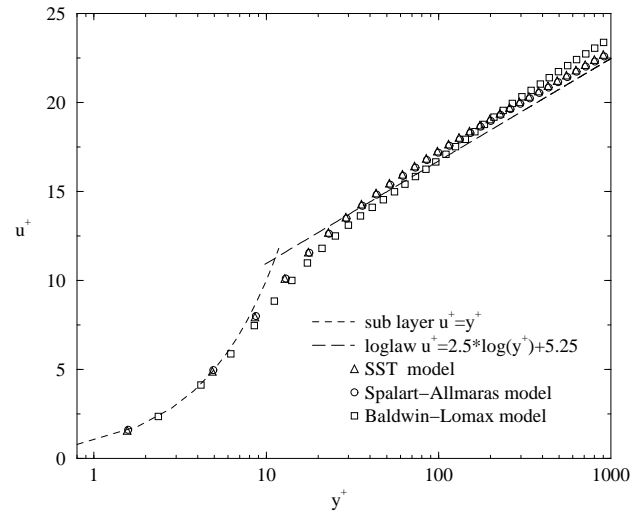


Fig. 1 Comparison of numerical turbulent velocity profile with theoretical data on a flat plate

Turbulent Flow over a Flat Plate

Air flow over a flat plate is chosen as a preliminary test case to validate the turbulence model implementations. The physical dimension of the computational domain is 1m by 0.5m by 0.005m in the streamwise direction (x), vertical direction (y), and transverse direction (z), respectively. The grid contains 128 cells in x , 80 cells in y and 1 cell in z . In order to resolve the thin boundary layer near the plate, exponentially distributed grid points resulting in a finer mesh near the plate are applied in the vertical direction, with the first grid point above the plate being at the distance of 4.7×10^{-6} m. The free stream Mach number in this simulation is taken to be 0.3. Theoretically, a turbulent velocity profile will consist of a linear viscous sublayer region close to the wall, transitioning through a buffer region to logarithmic behavior in the outer region. Fig. 1 shows the computational results from three turbulence models overlaid on the two theoretical curves for the viscous sublayer and the logarithmic layer where the axis of this plot are the nondimensional distance y^+ and the nondimensional velocity $u^+ = u/u_\tau$. The figure shows the good agreement of computational results obtained from Baldwin-Lomax, Spalart-Allmaras, and SST models with theoretical predictions.

Transonic Flow over the ONERA M6 Wing

The simulation of transonic viscous flow is benchmarked using the classic ONERA M6⁴⁴ 3D test case. A generalized grid is used to discretize the space around the wing, generated with a topologically adaptive mesh generation algorithm.^{45,46} In this technique, an unstructured surface mesh is advanced to create layers of cells. Edges are inserted or deleted to maintain a high quality mesh (for example, inserting new edges in convex regions to prevent the surface mesh from becoming too anisotropic). The insertion and deletion of edges yields general polyhedral elements and thus requires a generalized solver, such as the present one.

For this benchmark, one measured case of transonic

flow is employed, where the freestream Mach number is given by $M_\infty = 0.8395$ at an angle-of-attack of $\alpha = 3.06$. This case is viscous with a Reynolds number of $Re = 11.72 \times 10^6$ based on a mean chord length of 0.64607 meters. A freestream temperature of $T_\infty = 256\text{K}$ and pressure $p_\infty = 80,795\text{Pa}$ were used to achieve these conditions. A grid convergence study was performed by solving this case on three meshes. The coarsest grid was generated from 14K surface triangles to create a volume grid consisting of 1.1M cells. The finest grid was generated from a 32K surface triangles to create a volume grid consisting of 2.3M cells. The average y^+ for these grids was approximately 0.5. The SST turbulence model was employed, and the Sutherland formula was used for molecular viscosity (note that the Spalart-Allmaras model for the fine grid was also run, with results nearly identical to those presented here). The solution was run with a maximum CFL set to 30,000, a maximum time step set to 10^{-4} seconds, and $\mu_{relax} = 0.1$. The solver was run to engineering convergence in 2,500 time steps.

Since there is some unsteadiness in the problem, it is not possible to drive the residuals to machine zero. Instead, the behavior of integrated properties was used to determine when the solution had converged (for example, the total integrated momentum is illustrated in Fig. 2, and the integrated lift force is shown in Fig. 3). After 2,500 iterations each of these measures (and others) were unchanged to at least three significant figures. In addition, the fine grid solution was run for an additional 2,500 iterations with no noticeable changes in the C_p curves, giving further confidence in the convergence of the results.

The results are compared to experimental pressure measurements taken at seven span stations, as illustrated in Figs. 4 to 10. The first, 20% down the length of the wing, shows relatively good agreement, with a slight lag in the prediction of the shock on the upper surface. It is suspected that this difference may be due to the slightly coarser mesh in this region, and may also be attributed to an inappropriate application of a reflecting wall boundary condition where wind tunnel interference may play a role. Nonetheless, grid insensitivity is demonstrated, with the solutions on all grids nearly identical. The stations shown in Figs. 5 to 9 show excellent agreement between all three grids and experimental results, including capturing the details of the shock intersection, as can be seen in the C_p plots for $y/b = 0.80$ and $y/b = 0.90$. The last station shown in Fig. 10 shows good agreement at the leading edge with a divergence in simulated pressures at the upper trailing section. However, this difference is similar to other results found in the literature for this case.^{34,47}

In addition, it should be noted that this test case shows the effectiveness of the generalized grid strategy. Even the coarse grid for this case resolved most of the shock locations reasonably well with a grid cell count on the order of 1M cells. Published results from the *Cobalt*₆₀ code³⁴ for unstructured grid solutions on this case did not capture the features as well as the present results, in spite of us-

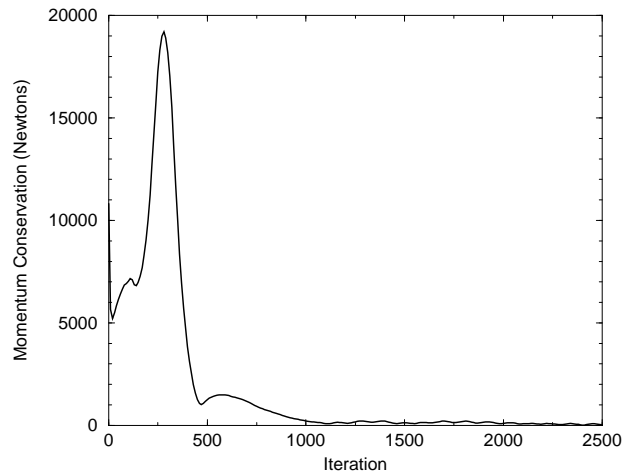


Fig. 2 Integrated Momentum v.s. Time step

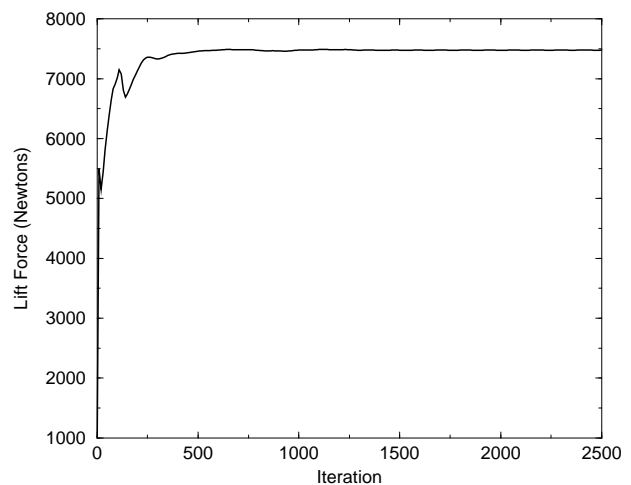


Fig. 3 Lift Force v.s. Time step

ing a 3M-cell unstructured mesh and similar algorithms. While not conclusive, indications are that the generalized grid used here was more effective at capturing flow details with less overall cost.

Hydrogen-Oxygen Combustion in an Experimental RBCC Engine

It is known that a propulsion system using ram-jet and scram-jet cycles must have another means of acceleration from rest to low supersonic speeds, at which point the ram-jet cycle can generate sufficient thrust for further acceleration. Rocket-based combined-cycle (RBCC – i.e. single duct air augmented ram-jet/dual mode scram-jet) propulsion systems use rocket motors to accomplish this, and are characterized by a high degree of integration between rocket and ram-jet. The RBCC engine offers higher engine thrust-to-weight ratios than competing air-breathing engines, while maintaining an I_{sp} advantage over rocket engines. As a result, there has been recent interest in RBCC engines for future space transport vehicles. In this section, the application of the CHEM code in simulating an experimental RBCC engine is detailed; it should be noted that

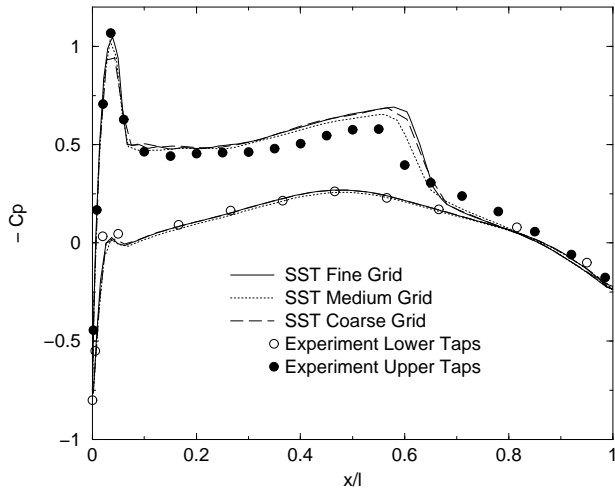


Fig. 4 ONERA-M6 Wing, C_p at $y/b=0.20$

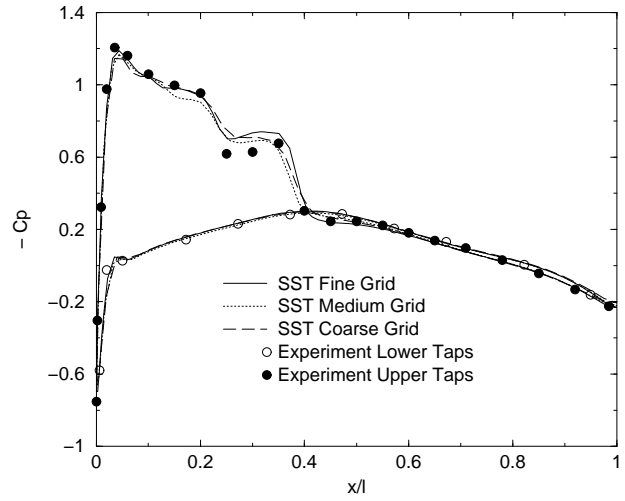


Fig. 7 ONERA-M6 Wing, C_p at $y/b=0.80$

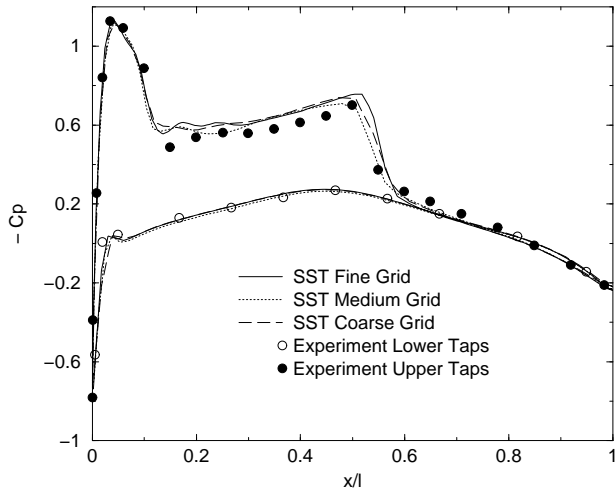


Fig. 5 ONERA-M6 Wing, C_p at $y/b=0.44$

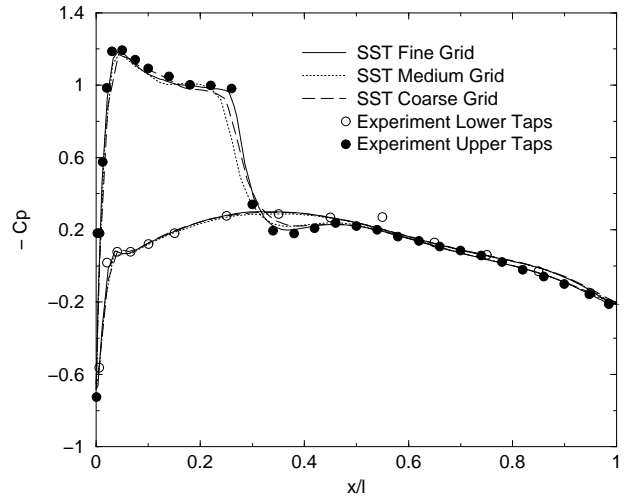


Fig. 8 ONERA-M6 Wing, C_p at $y/b=0.90$

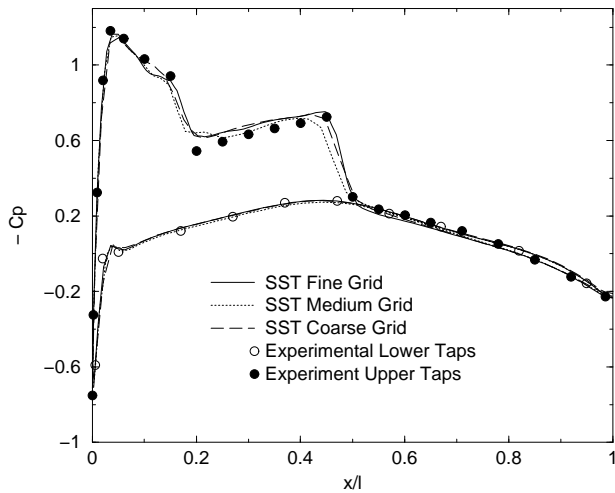


Fig. 6 ONERA-M6 Wing, C_p at $y/b=0.65$

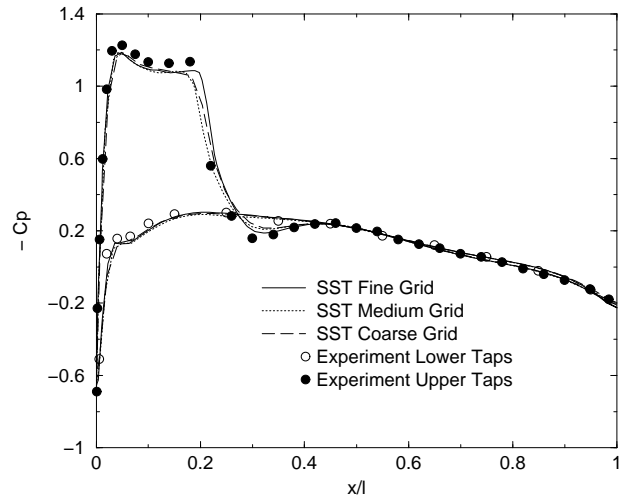


Fig. 9 ONERA-M6 Wing, C_p at $y/b=0.95$

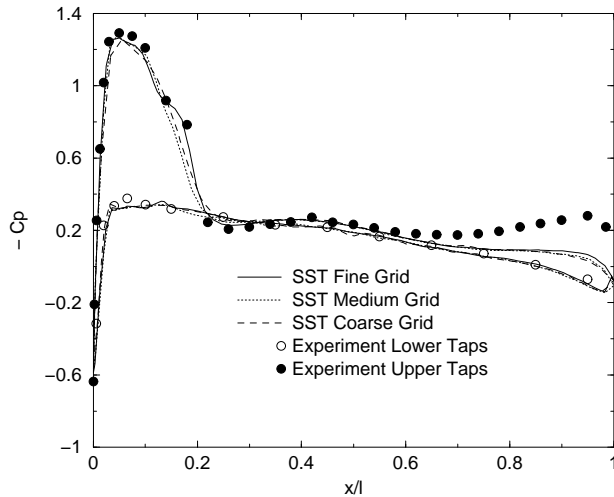


Fig. 10 ONERA-M6 Wing, C_p at $y/b=0.99$

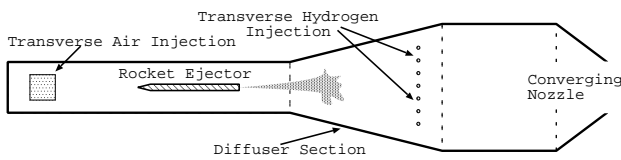


Fig. 11 Schematic of PSU-RBCC Experiment

this geometry was specifically designed to validate RBCC simulation codes. Only one case is included here as an example of the capabilities of the algorithm.

The experiment was performed at the Pennsylvania State University (PSU) Propulsion Engineering Research Center, and Case 4 of the Direct Connection studies is described here.⁴⁸ The PSU-RBCC 500DC series (Direct Connections) feature a closed front end and with transverse air inlets, as illustrated in the schematic shown in Fig. 11. An ejector rocket is placed further down the rectangular duct just before a diffuser section. At this point gaseous hydrogen is transversely injected into the duct. Combustion occurs downstream of the injection. Finally, a converging nozzle accelerates the flow at the exit. The computational domain begins at the left end of the air inlet (-0.508 m), and ends beyond the exit of the nozzle (2.3622 m). The total length of the computational domain is 2.8702 m (113 inches). The RBCC engine inlet plane is located at $x = 0$, where the thruster nozzle exhaust plume and ram-jet inlet air flow into the duct.

An unstructured grid for a quarter symmetry of this RBCC duct was generated using SolidMesh, by way of the `aflr3`⁴⁹ grid generation software, with prismatic elements in viscous boundary layers near walls and tetrahedral isotropic elements in the interior volume. The final grid used in the simulations presented here consist of 3.4 million elements, with packing near the rocket ejector plume and transverse hydrogen injection regions.

For Case 4 of the Direct Connection study a 0.9145kg/s mass flow of air injection is measured. The ejector rocket was provided with 0.034473kg/s of GH_2 and 0.2758kg/s of GO_2 . The downstream hydrogen injection accounted for

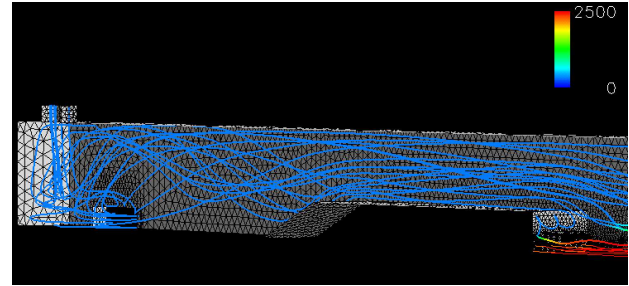


Fig. 12 Forced Air Inlet Streamlines superimposed with temperatures

0.02676kg/s of mass flow.

The computational simulation was performed in a total of 5,000 iterations with mass, momentum, and energy residuals reduced by at least three orders of magnitude. The first 1,000 iterations were performed without the transverse hydrogen injection, then the transverse hydrogen injection was activated for the final 4,000 iterations. The simulation was run with a maximum CFL set to 1×10^6 , with a maximum time step set to 5×10^{-5} seconds before transverse hydrogen injection and 1×10^{-5} after hydrogen injection. A setting of $\mu_{relax} = 0.02$ was used to maintain stability. A y^+ of about unity was maintained through most of the duct. Hydrogen-oxygen combustion was modeled using a 7-species, 32-reaction mechanism from Evans and Schexnayder.⁵⁰ Thermodynamic properties were derived from vibrational equilibrium, and transport properties were obtained from CHEMKIN. The SST turbulence model was employed. A turbulent Prandtl number of 0.9 and a turbulent Schmidt number of 0.7 were used to model turbulent transport effects. No corrections for turbulent chemistry source terms were included in the model.

Fig. 12 shows the streamlines for the forced air inlet region of the RBCC duct. In this figure, the streamlines are colored by fluid temperatures. As it can be seen, a complex flow pattern is captured involving intersecting jets of air. Recirculation and entrainment is also clearly seen at the thruster exit. The downstream hydrogen injection is illustrated in Fig. 13. Here, the air-inlet streamlines are thick, while the thruster streamlines are thin. These streamlines and the walls are colored by fluid temperatures. The hydrogen injection streamlines are colored white to illustrate the entrainment of the hydrogen jet with the flow. The symmetry planes have contours of H_2 mass fractions, with the maximum value (in red) being 17% and low value of 0% (in blue). From the streamlines it is evident that the majority of the injected air travels through the center of the RBCC duct at the point of transverse hydrogen injection. This is further confirmed by the relatively large mass fractions of H_2 observed at the symmetry plane, where it is suspected that this fuel rich condition is likely due to an insufficient air mass-flow rate in that region. In general, these figures illustrate a highly complex flow field with strongly non-linear and stiff source terms. It should be stressed that these results were obtained with relatively large time steps and few iterations.

Experimental measurements of this configuration con-

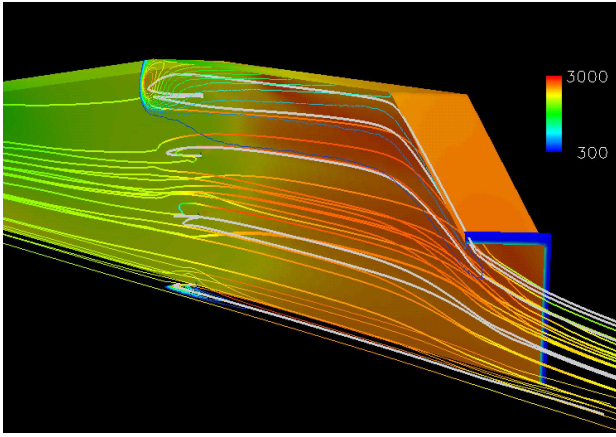


Fig. 13 H2 Injection in RBCC Afterburner section

sisted of pressure measurements recorded along the side and top walls of the RBCC duct, while temperature and species measurements were performed using a window at $x = 0.5969\text{m}$ using Raman imaging. For each firing of the rocket, static pressure is recorded along the air duct walls. Sixteen channels are located on both the top and side wall of the duct. The comparison of measured to experimental pressures are shown in Fig. 14. The predicted wall pressures are within 2% of the experimental results.

Fig. 15 shows temperature comparisons with experimental measurements by PSU at $x = 0.5969\text{m}$. Near the centerline ($y = 0$) the predicted temperature is about 200K higher than the measured one, at $y = 0.03\text{m}$, the measured and computed profiles are better, but still appear to overestimate temperature. However, these results appear to be within the range of variation observed in the experimental results. The exact cause for the higher predicted temperatures is unknown, but may be due to the lack of including turbulence effects in the chemistry source terms. In effect, the model is probably predicting faster combustion than what is physically meaningful. Fig. 16 shows comparisons of species mass fraction distributions with measurements at $x = 0.5969\text{m}$. The predicted mass concentrations agree well with the measured ones.

Overall this simulation provides very encouraging results, particularly when one considers the considerable complexity and uncertainty involved in properly modeling the physics of this problem. More work needs to be done to understand sensitivity of these results to combustion and turbulent chemistry models. This case is part of a more intensive study that will seek to answer those questions. However, these results are presented here to demonstrate the effectiveness and robustness of CHEM when solving highly complex and stiff numerical equations. In this regard, the solver performed extremely well, allowing simulations using relatively large time steps, particularly when considering the diverse time and space scales involved in this case.

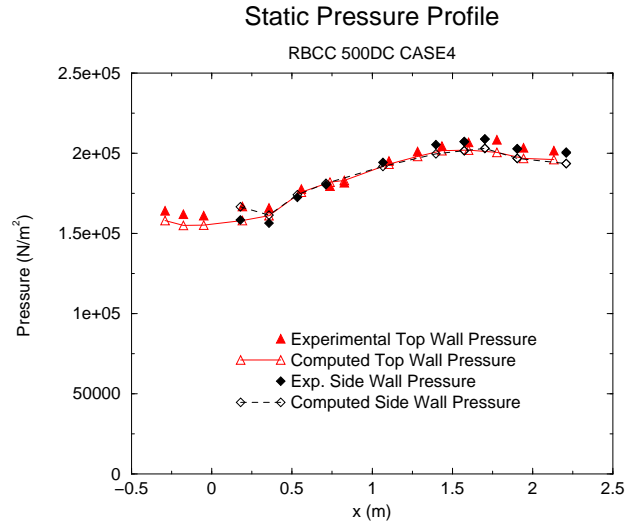


Fig. 14 Wall Pressures experiment and simulation comparison

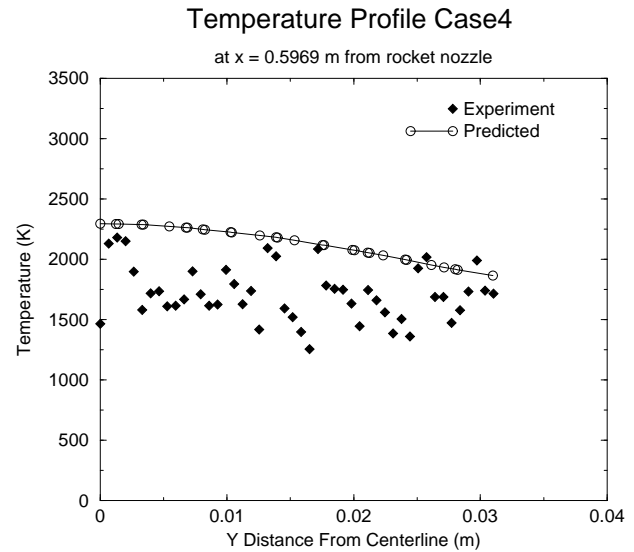


Fig. 15 Temperature, experiment and simulation comparison

Conclusions

A general software system that supports the development of flexible multidisciplinary simulation models was introduced. The work described in this paper is one case study in the effectiveness of this system in the development of practical numerical models for complex engineering problems. In particular, the development of a detailed model for chemically reacting flowfields using a generalized finite-volume discretization was detailed. New methods for improving the robustness of such solvers have been mentioned: in particular, the local time stepping scheme and Jacobian smoothing techniques have been found to be particularly effective in the solution of the highly stiff equations that arise from computational combustion problems. The effectiveness of these strategies has been shown on both non-reacting and reacting benchmark problems. Specifically, large time-step and CFL conditions have been used in the solution of complex flow fields involving hydrogen-

Species's Mass Fraction Profile: Case 4

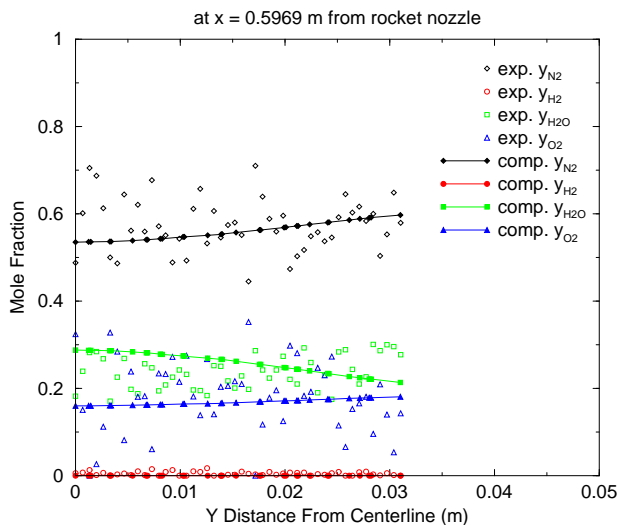


Fig. 16 Species mass fractions, experiment and simulation comparison

oxygen combustion, allowing for engineering accuracy to be achieved expeditiously.

Acknowledgments

This work was partially supported by the National Science Foundation, which established the Engineering Research Center (now ERC) at Mississippi State University. Additional support was provided by NSF through grant ITR-0085969 and by NASA Marshall Space Flight Center and Stennis Space Flight Center through grant NAS13-98033 (delivery order no. 161). In particular, we would like to thank our NASA technical monitor, Jeff West, for his instructive and helpful comments.

References

- ¹Cinnella, P. and Grossman, B., "Computational Methods for Chemically Reacting Flows," *Handbook of Fluid Dynamics and Fluid Machinery*, Wiley and Sons, New York, 1996, pp. 1541–1590.
- ²Luke, E. A., *A Rule-Based Specification System for Computational Fluid Dynamics*, Ph.D. thesis, Mississippi State University, 1999.
- ³Luke, E., "Loc: A Deductive Framework for Graph-Based Algorithms," *Third International Symposium on Computing in Object-Oriented Parallel Environments*, edited by S. Matsuoka, R. Oldehoeft, and M. Tholburn, No. 1732 in Lecture Notes in Computer Science, Springer-Verlag, December 1999, pp. 142–153.
- ⁴Kingsley, G., J. M. Siegel, J., Harrand, V. J., Lawrence, C., and Luker, J. J., "Development of a multi-disciplinary computing environment (MIDICE)," *7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, No. AIAA-98-4738, St. Louis, MO, September 1998, pp. 251–260.
- ⁵Lytle, J. K., Follen, G. J., Naiman, C. G., Evans, A. L., Veres, J., and Lopes, P., "1999 Numerical Propulsion System Simulation Industry Review," Tech. Rep. TR 209795, NASA, September 2000.
- ⁶Budge, K. G. and Peery, J. S., "Experiences Developing ALEGRA: A C++ Coupled Physics Framework," *SIAM Workshop on Object Oriented Methods for Interoperable Science and Engineering Computing*, IBM TJ Watson Research Center, New York, October 1998.
- ⁷Haney, S., Crottinger, J., Karmesin, S., and Smith, S., "PETE, the Portable Expression Template Engine," *Dr. Dobbs Journal*, October 1999.
- ⁸Veldhuizen, T. L., "Arrays in Blitz++," *Proceedings of the 2nd International Scientific Computing in Object-Oriented Parallel Environments*

(ISCOPE'98), Lecture Notes in Computer Science, Springer-Verlag, 1998.

⁹Cross, M., Chow, P., Bailey, C., Croft, N., Ewer, J., Leggett, P., McManus, K., Pericleous, K. A., and Patel, M. K., "PHYSICA - A Software Environment for the Modelling of Multi-Physics Phenomena," *Proceedings of ICIAM*, July 1995.

¹⁰Luke, E. A., Koomullil, R., and Soni, B. K., "Integrated Multi-disciplinary Simulation Environment for Analysis and Testing of RBCC Systems," Tech. rep., Simulation and Design Center, Mississippi State University, 2 Research Blvd, Starkville, MS 39759, June 2003, Final Report for NASA grant NAS13-98033 Delivery Order #161.

¹¹Vincent, W. G. and Kruger, C. H., *Introduction to Physical Gas Dynamics*, Krieger Publishing Company, 1965.

¹²"NIST-JANAF Thermochemical Tables, 4th edition," Malcolm W. Chase, Jr. Washington, DC: American Chemical Society ; New York: American Institute of Physics for the National Institute of Standards and Technology, 1998.

¹³"NIST Chemistry WebBook," <http://webbook.nist.gov>, March 2003, Database Number 69.

¹⁴Gupta, R. N., Yos, J. M., Thompson, R. A., and Lee, K. P., "A Review of Reaction Rates and Thermodynamic and Transport Properties for an 11-Species Air Model for Chemical and Thermal Nonequilibrium Calculation to 30,000K," Tech. rep., NASA Reference Publication 1232, 1990.

¹⁵Kee, R., Rupley, F., and Miller, J., "Chemkin II: A fortran chemical kinetic package for modeling well-stirred reactors," Tech. Rep. SAND 89-8009B, Sandia National Laboratories, 1989.

¹⁶Baldwin, B. and Lomax, H., "Thin Layer Approximation and Algebraic Model for Separated Turbulent Flows," *Proceedings of the AIAA 16th Aerospace Sciences Meeting*, AIAA, January 1978, AIAA-78-257.

¹⁷Spalart, P. R. and Allmaras, S. R., "A One-equation Turbulence Model for Aerodynamic Flows," *AIAA-92-0439*, 1992.

¹⁸Menter, F. R., "Influence of Freestream Values on $k - \omega$ Turbulence Model Predictions," *AIAA Journal*, Vol. 30, No. 6, June 1992, pp. 1657–1659.

¹⁹Wilcox, D. C., "Reassessment of the Scale-Determining Equation for Advanced Turbulence Models," *AIAA Journal*, Vol. 26, No. 11, 1988, pp. 1299–1310.

²⁰Wilcox, D. C., "A Half Century Review of the $k - \omega$ Model," Tech. rep., AIAA, 1991, AIAA 91-1784.

²¹Menter, F. R., "Two-Equation Eddy-Viscosity Turbulence Models for Engineering Applications," *AIAA Journal*, Vol. 32, No. 8, August 1994, pp. 1598–1605.

²²Wilcox, D. C., *Turbulence Modeling for CFD*, DCW Industries, 1998.

²³Johnson, D. A. and King, L. S., "Mathematically Simple Turbulence Closure Model for Attached and Separated Turbulent Boundary Layers," *AIAA Journal*, Vol. 23, No. 11, November 1985, pp. 1684–1692.

²⁴Samimy, M. and Elliott, G. S., "Effects of Compressibility on the Characteristics of Free Shear Layers," *AIAA Journal*, Vol. 28, No. 3, March 1990, pp. 439–445.

²⁵Sarkar, S. and Lakshmanan, B., "Application of a Reynolds Stress Turbulence Model to the Compressible Shear Layer," *AIAA Journal*, Vol. 29, No. 5, May 1991, pp. 743–749.

²⁶Koomullil, R. and Soni, B., "Flow Simulation Using Generalized Static and Dynamic Grids," *AIAA Journal*, Vol. 37, No. 12, December 1999, pp. 1551–1557.

²⁷Hansen, A. G., "Generalized Control Volume Analyses with Application to the Basic Laws of Mechanics and Thermodynamics," *Bulletin of Mechanical Engineering Education*, Vol. 4, 1965, pp. 161–168.

²⁸Thomas, P. D. and Lombard, C. K., "Geometric Conservation Law and Its Application to Flow Computations on Moving Grids," *AIAA Journal*, Vol. 17, No. 10, 1978, pp. 1030–1037.

²⁹Barth, T. J. and Jespersen, D. C., "The Design and Application of Upwind Schemes on Unstructured Meshes," Tech. rep., AIAA, 1989, AIAA 89-0366.

³⁰Venkatkrishnan, V., "On the accuracy of limiters and convergence to steady state solutions," Tech. rep., AIAA, 1993, AIAA 93-0880.

- ³¹Walters, R. W., Cinnella, P., Slack, D. C., and Halt, D., "Characteristic-Based Algorithms for Flows in Thermodynamic Nonequilibrium," *AIAA Journal*, Vol. 30, 1992, pp. 1304–1313.
- ³²Quirk, J., "A contribution to the great Riemann debate," Tech. rep., ICASE Report, 1992.
- ³³Einfeldt, B., "On Godunov-type method for gas dynamics," *SIAM Journal on Numerical Analysis*, Vol. 25, 1988, pp. 294–318.
- ³⁴Strang, W., Tomaro, R., and Grismer, M., "The Defining Methods of Cobalt60: A Parallel, Implicit, Unstructured Euler/Navier-Stokes Flow Solver," AIAA, 1999, Paper Number 99-0786.
- ³⁵Beam, R. and Warming, R., "An Implicit Factored Scheme for the Compressible Navier-Stokes Equations," *AIAA Journal*, Vol. 16, No. 4, April 1978, pp. 393–404.
- ³⁶Ridders, C. J. F., "A New Algorithm for Computing a Single Root of a Real Continuous Function," *IEEE Transactions on Circuits and Systems*, Vol. 26, 1979, pp. 979–980.
- ³⁷Merci, B., Steelant, J., Vierendeels, J., Rienslagh, K., and Dick, E., "Computational Treatment of Source Terms in Two-Equation Turbulence Model," *AIAA Journal*, Vol. 38, 2000, pp. 2085–2093.
- ³⁸Liu, Q., Luke, E. A., Cinnella, P., and Tang, L., "Coupling Heat Transfer and Fluid Flow Solvers for Multi-Disciplinary Simulations," 2004, submitted.
- ³⁹Tong, X., Luke, E. A., and Tang, L., "Evaluations of the Shear-Stress Transport Turbulence Model for Heat Transfer Applications," *Proceedings of the 42nd AIAA Aerospace Sciences Meeting and Exhibit*, AIAA, January 2003, AIAA 2003-0769.
- ⁴⁰Wu, J., Tang, L., Tong, X.-L., Luke, E., and Cinnella, P., "A Comprehensive Numerical Study of Jet Flow Impingement over Flat Plates at Varied Angles," *Journal of Spacecraft and Rockets*, Vol. 23, No. 1, May-June 2002, pp. 357–366.
- ⁴¹Luke, E. A., Tong, X., Wu, J., Tang, L., and Cinnella, P., "A Step Towards 'Shape-Shifting' Algorithms: Reacting Flow Simulations Using Generalized Grids," *Proceedings of the 39th AIAA Aerospace Sciences Meeting and Exhibit*, AIAA, January 2001, AIAA-2001-0897.
- ⁴²Wu, J., Tang, L., Luke, E. A., Tong, X.-L., and Cinnella, P., "Numerical Simulation of Under-Expanded Jet Impingement," *Proceedings of the SECTAM-XX*, edited by H. V. Tippur and P. K. Raju, Vol. 20, April 2000, Paper No. FM-20.
- ⁴³Tong, X.-L., Luke, E. A., and Cinnella, P., "Numerical simulations of chemically reactive turbulent flows using the Loci system," *Proceedings of the SECTAM-XX*, edited by H. V. Tippur and P. K. Raju, Vol. 20, April 2000, Paper No. FM-19.
- ⁴⁴Schmitt, V. and Charpin, F., "Pressure Distributions on the ONERA-M6-Wing at Transonic Mach Numbers," Tech. Rep. AGARD Report No. AR 138, May 1979.
- ⁴⁵Chalasani, S., Thompson, D., and Soni, B., "Topological Adaptivity for Mesh Quality Improvement," *Proceedings of the 8th International Conference on Numerical Grid Generation in Computational Field Simulations*, Honolulu, HI, June 2002, p. Available on CD.
- ⁴⁶Chalasani, S. and Thompson, D., "Quality Improvements in Extruded Meshes using Topologically Adaptive Generalized Elements," *International Journal of Numerical Methods Engineering*, (submitted).
- ⁴⁷Mani, M., Ladd, J., Cain, A., and Bush, R. H., "An Assessment of One- and Two-Equation Turbulence Models for Internal and External Flows," AIAA, 1997, Paper Number 97-2010.
- ⁴⁸Santorio, R. and Pal, S., "Experimental and Analytical Modeling of the Rocket Ejector Mode of a Combined Cycle Rocket-Based Engine," Tech. rep., Propulsion Engineering Research Center, Pennsylvania State University, University Park, PA, 16802, June 2001, Final Report for NASA Contract Grant NAS8-40890.
- ⁴⁹Marcum, D. and Weatherill, N. P., "Unstructured Grid Generation Using Iterative Point Insertion and Local Reconnection," *AIAA Journal*, Vol. 33, No. 9, September 1995, pp. 1619–1625.
- ⁵⁰Evans, J. S. and C. J. Schexnayder, J., "Influence of Chemical Kinetics and Unmixidness on Burning in Supersonic Hydrogen Flames," *AIAA Journal*, Vol. 18, No. 2, 1979, pp. 188–193.