Comprehensive Code Verification Techniques for Finite Volume CFD Codes

Subrahmanya P. Veluri[1] and Christopher J. Roy[2]
Aerospace and Ocean Engineering Department
Virginia Tech, Blacksburg, Virginia 24061

Edward A. Luke[3]
Computer Science and Engineering Department
Mississippi State University, Starkville, MS 39762

A detailed code verification study of a finite volume Computational Fluid Dynamics (CFD) code using the Method of Manufactured Solutions is presented. The correctness of the code is verified through order of accuracy testing. Systematic mesh refinement required for order of accuracy testing and the way it is achieved particularly for unstructured meshes is discussed. The verification testing is performed on different mesh types which include triangular and quadrilateral elements in 2D and tetrahedral, prismatic, and hexahedral elements in 3D. The sensitivity of the order of accuracy to both mesh quality and mesh topology is examined. Along with the baseline steady-state governing equations, transport models, turbulence models, boundary conditions, and unsteady flows are verified. A new approach for the combined verification of spatial and temporal terms in the governing equations is developed and assessed.

Keywords: Code Verification, Order of Accuracy, Manufactured Solutions

Nomenclature

C_p, C_v = specific heats, J/Kg · K

DE = discretization error

e = energy, J

[1] Graduate Research Assistant, Currently Test Engineer at ANSYS Inc.
[2] Associate Professor.
[3] Associate Professor.

g_p = coefficient of the leading error term

h = normalized grid spacing; enthalpy, J/Kg · K

h_f = heat of formation, J/Kg · K

F = blending function

f = general solution variable

k = turbulent kinetic energy

p = spatial order of accuracy; pressure, N/m^2

q_L = laminar heat flux

q_T = turbulent heat flux

R = gas constant, J/Kg · K

r = refinement factor

T = temperature, K

t = time, sec

V_i = Volume of cell 'i'

u, v, w = Cartesian velocity components, m/s

x, y, z = Cartesian coordinates, m

Greek Symbols

ε = error

μ = viscosity, $N \cdot sec/m^2$

μ_T = turbulent viscosity, $N \cdot sec/m^2$

ρ = density, Kg/m^3

ω = turbulent dissipation rate

Subscripts

exact = exact continuum value

k = mesh level, 1, 2, 3, etc.; fine to coarse

ref = reference value

1. Introduction

VERIFICATION addresses the mathematical correctness of the simulations. There are two fundamental aspects to verification: code verification and solution verification [1-3]. Code verification is the process of ensuring, to the degree possible, that there are no mistakes (bugs) in a computer code or inconsistencies in the solution algorithm. Solution verification is the process of estimating the three types of numerical error that can occur in numerical simulations: round-off error, iterative error, and discretization error. This paper focuses mainly on code verification. Current practices in code verification verify that the observed order of accuracy of the discretization error asymptotically approaches the formal order of accuracy of the discretization scheme as the mesh is systematically refined.

One of the main difficulties in verifying a code is identifying exact solutions to the governing equations which exercise all terms. Traditional exact solutions exist only when the governing equations are fairly simple, which is certainly not the case for modern CFD codes which are expected to handle complex physics (turbulence, combustion, real gas effects, etc.), complex geometries, and significant nonlinearities. The Method of Manufactured Solutions, or MMS, is a general approach for obtaining exact solutions [1-3]. Rather than trying to find an exact solution to a system of partial differential equations, the goal is to "manufacture" an exact solution to a slightly modified set of equations. Order of accuracy verification is a rigorous code verification assessment which involves comparing the observed order of accuracy of the discretization error to the formal order of accuracy of the chosen numerical methods. The observed order of accuracy can fail to match the formal order due to mistakes in the computer code, defective numerical algorithms, solutions which are not sufficiently smooth, numerical solutions which are not in the asymptotic mesh convergence range [2], and large round-off or iterative error. The asymptotic range is defined as the range of discretization sizes (Δx, Δy, Δt, etc.) where the lowest-order terms in the truncation error dominate.

The first application of MMS for code verification was by Roache and Steinberg in 1984 [4]. In their pioneering work, they used the MMS approach to verify a code for generating three-dimensional transformations for elliptic partial differential equations. Additional discussions of the MMS procedure for code verification have been presented by Roache [1, 5]. The book by Knupp and Salari [6] is a comprehensive discussion of code verification, MMS, and order of accuracy verification. More recently, Oberkampf and Roy [3] discussed the use of MMS for generating exact solutions along with the order of accuracy testing for code verification.

In prior work, MMS has been used to verify two compressible CFD codes [7]: Premo [8] (developed by Sandia National Laboratories) and WIND [9] (developed by the NPARC alliance). In Ref. 7, the authors successfully verified both the inviscid Euler equations and the laminar Navier-Stokes equations; however, this study employed only Cartesian meshes. An alternative statistical approach to MMS was proposed by Hebert and Luke [10] for the Loci-CHEM combusting CFD code [11]. In their approach, they employ a single mesh level which is shrunk down (thus providing a locally refined mesh) and used to statistically sample the discretization error in different regions of the domain of interest. Their work successfully verified the Loci-CHEM CFD code for the 3D, multi-species, laminar Navier-Stokes equations using both statistical and traditional MMS. Another similar approach to statistical MMS is the downscaling approach to order verification [12, 13] which also employs a single mesh which is scaled down about a single point in the domain instead of statistically sampling the smaller meshes in the domain of interest and hence eliminating the statistical convergence issues associated with the statistical MMS approach. Both the statistical MMS and the downscaling approaches have an advantage of being relatively inexpensive because they do not require mesh refinement. But one of the disadvantages of using these methods is that they neglect the possibility of discretization error transport into the scaled-down domain. Thus, it is possible to pass a code order of accuracy verification test using statistical MMS or the downscaling approach for a case that would fail the test using traditional MMS.

There have been some coordinated efforts to apply MMS to turbulent flows. Pelletier and co-workers have summarized their work on 2D incompressible turbulent shear layers using a finite element code with a focus on a logarithmic form of the k-ε two-equation RANS model in Refs. 14 and15. They employed Manufactured Solutions which mimic turbulent shear flows, with the turbulent kinetic energy and the turbulent eddy viscosity as the two quantities specified in the Manufactured Solution. For the cases examined, they were able to verify the code by reproducing the formal order of accuracy of the code. More recently, Eca and co-workers have published a series of papers on Manufactured Solutions for the 2D incompressible turbulent Navier-Stokes equations [16-18]. They also employed physically-based Manufactured Solutions, in this case mimicking wall-bounded turbulent flow. This group looked at both finite-difference and finite-volume discretizations, and examined a number of turbulence models including the Spalart-Allmaras one-equation model [19] and two two-equation models: Menter's baseline (BSL) version k-ω model [20] and Kok's turbulent/non-turbulent k-ω model [21]. While successful in some cases, their physically-relevant Manufactured Solution often led to numerical instabilities, a reduction in the observed

mesh convergence rate, or even inconsistency of the numerical scheme (i.e., the discretization error did not decrease as the mesh was refined). In order to independently test different aspects of the governing equations, in some cases they replaced certain discretized terms (or even whole equations) with the analytic counterpart from the Manufactured Solution. For the Spalart-Allmaras model they specified the working variable $\tilde{\nu}_t$, while for the two equation models they specified both the turbulent eddy viscosity and the turbulent kinetic energy. The cases they examined employed a Reynolds number of 10^6 and used Cartesian meshes which were clustered in the y-direction towards the wall.

Our approach to code verification for RANS models differs from the previous work in a number of ways. While the above work focused on physically-based exact solutions with complex exponentials to mimic the turbulence quantities found in real turbulent flows, we simply use sinusoidal functions. Our argument for taking this approach is that the goal of code verification is to perform mathematical tests to ensure the discretization approach and the implementations into a code do in fact match the original governing partial differential equations and their solution [22, 23]. In order to verify the implementation of boundary conditions, the manufactured solution must be tailored to exactly satisfy a given boundary condition on a domain boundary and Bond et al. [24] developed a general approach for doing the same.

2. Governing Equations

The verification concepts from this work can be applied to any finite volume code, but the CFD code on which verification is performed in the current work is Loci-CHEM [11, 25]. Loci-CHEM was developed at Mississippi State University using the Loci framework [26, 27] and can simulate three-dimensional flows of turbulent, chemically-reacting mixtures of thermally perfect gases. The Loci framework provides a high-level programming environment for numerical methods that is automatically parallel and utilizes a logic-based strategy to detect or prevent common software faults (such as errors in loop bounds or errors caused by subroutine calling sequences being inconsistent with data dependencies). The baseline governing equations include the 3D Euler and Navier-Stokes equations [28].

3. Method of Manufactured Solutions

3.1 Methodology

The most rigorous approach to code verification is the order of accuracy test [1, 6], which determines whether or not the discretization error is reduced at the expected rate. This test is equivalent to determining whether the observed order of accuracy matches the formal order of accuracy. For all discretization approaches (finite difference, finite volume, finite element, etc.) the formal order of accuracy is usually obtained from a truncation error analysis of the discrete algorithm. The truncation error analysis appears to be sufficient for regular, structured grids; however, it has been shown to under-predict the formal order of solution convergence for finite volume methods on unstructured meshes [29]. The observed order of accuracy is directly computed from code output for a given set of simulations on systematically refined grids.

The discretization error is formally defined as the difference between the exact solution to the discrete equations and the exact solution to the governing partial differential equations. Since the exact solution to the discrete equations (which will be different on different mesh levels) is generally not known, the numerical solution is obtained by solving the discrete residual equations to machine zero levels neglecting iterative and round-off error. The observed order of accuracy can be evaluated either locally within the solution domain or globally by employing a norm of the discretization error. While we have examined L_1, L_2, L_∞ norms of the current code verification study, here we report only L_2 norms for brevity. The volume weighted L_2 norm for mesh level k is defined as

$$L_{2,k} = \left[\sum_{i=1}^{N} \left[V_i \left(DE_{k,i} \right)^2 \right] \Big/ \sum_{i=1}^{N} V_i \right]^{1/2} \tag{1}$$

where the *i* index denotes a cell center value of one of the conserved variables and V_i is the volume of cell *i*.

Consider a series expansion of the discretization error in terms of h_k, a measure of the element size on the mesh level k,

$$DE_k = f_k - f_{exact} = g_p h_k^p + HOT \tag{2}$$

where f_k is the numerical solution on mesh k, g_p is the coefficient of the leading error term, and *p* is the formal order of accuracy. Neglecting the higher order terms, we can write the discretization error equation for a fine mesh (k=1) and a coarse mesh (k=2) in terms of the observed order of accuracy $\hat{p}$ as

$$DE_1 = f_1 - f_{exact} = g_p h_1^{\hat{p}} \quad \text{and} \quad DE_2 = f_2 - f_{exact} = g_p h_2^{\hat{p}} \tag{3}$$

Since the exact solution is known, these two equations can be solved for the observed order of accuracy $\hat{p}$. Introducing r, the ratio of coarse to fine mesh element spacing ($r=h_2/h_1$), the observed order of accuracy becomes

$$\hat{p} = \ln\left(\frac{DE_2}{DE_1}\right) \Big/ \ln(r) \tag{4}$$

Thus, when the exact solution is known, only two solutions are required to obtain the observed order of accuracy.

When evaluating the observed order of accuracy, round-off and iterative convergence error can adversely affect the results. Round-off error occurs due to finite digit storage on digital computers. Iterative error occurs any time an iterative method is used, as is generally the case for nonlinear systems and large, sparse linear systems. The discretized form of nonlinear equations can necessarily be solved to within machine round-off error; however, in practice, the iterative procedure is usually terminated earlier to reduce computational effort. In order to ensure that these source of error do not adversely impact the observed order of accuracy calculation [2], both round-off and iterative error should be at least 100 times smaller than the norm of the fine mesh discretization error (i.e., $<0.01\times DE_1$). For all cases presented herein, double precision computations are used and the iterative residuals are reduced down to machine zero. For the current computations, this corresponds to an iterative residual reduction of approximately 14 orders of magnitude.

The procedure for applying MMS with order of accuracy verification is demonstrated below for a simple example problem:

1. Choose the specific form of governing equations; here we examine the 1D unsteady heat equation

$$\frac{\partial T}{\partial t} - \alpha \frac{\partial^2 T}{\partial x^2} = 0 \tag{5}$$

2. Choose the manufactured solution

$$T(x,t) = T_0 \exp(t/t_0)\sin(\pi x/L) \tag{6}$$

3. Operate the governing equations on the chosen solution, resulting in analytic source terms

$$\begin{aligned} \frac{\partial T}{\partial t} &= T_0 \sin(\pi x/L)\frac{1}{t_0}\exp(t/t_0) \\ \frac{\partial^2 T}{\partial x^2} &= -T_0 \exp(t/t_0)(\pi/L)^2 \sin(\pi x/L) \end{aligned} \tag{7}$$

4. Solve the modified governing equation (original equation plus source terms) on various mesh levels

$$\frac{\partial T}{\partial t} - \alpha \frac{\partial^2 T}{\partial x^2} = \left[\frac{1}{t_0} + \alpha \left(\frac{\pi}{L} \right)^2 \right] T_0 \exp(t/t_0) \sin(\pi x / L) \tag{8}$$

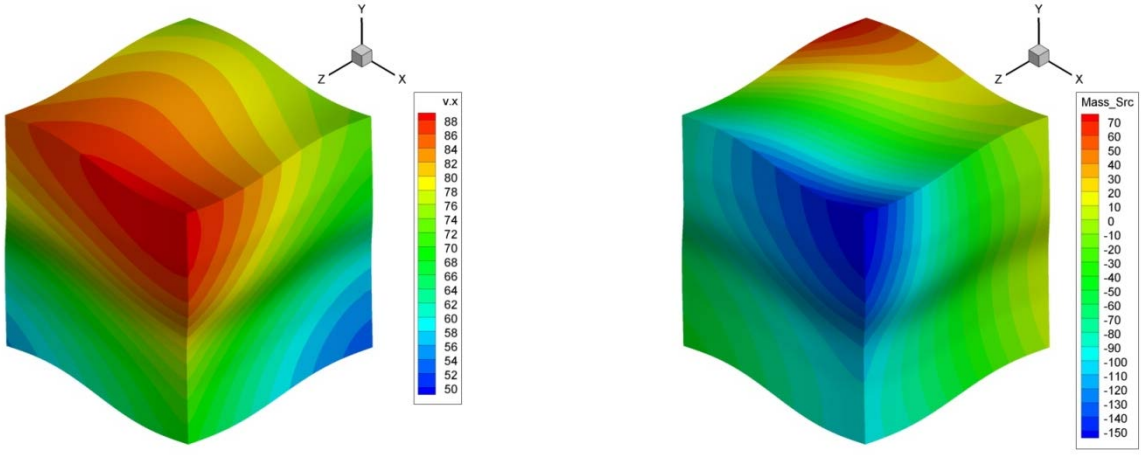
5. Compute the observed order of accuracy of the discretization error and compare it with the formal order of accuracy

3.2 Baseline Manufactured Solutions

In our current work, we adhere to the philosophy that code verification is simply a mathematical test to ensure the numerical solution truly represents the solution to the continuum mathematical equations that are being solved. As such, we have specifically chosen Manufactured Solutions which are not physically realistic, but which are simple, smooth, and exercise all terms in the governing equations. The steady Manufactured Solutions employed take the following form

$$\begin{aligned} \phi(x,y,z) = \phi_0 &+ \phi_x f_s\left(\frac{a_{\phi x} \pi x}{L} \right) + \phi_y f_s\left(\frac{a_{\phi y} \pi y}{L} \right) + \phi_z f_s\left(\frac{a_{\phi z} \pi z}{L} \right) + \phi_{xy} f_s\left(\frac{a_{\phi xy} \pi xy}{L^2} \right) \\ &+ \phi_{yz} f_s\left(\frac{a_{\phi yz} \pi yz}{L^2} \right) + \phi_{zx} f_s\left(\frac{a_{\phi zx} \pi zx}{L^2} \right) \end{aligned} \tag{9}$$

where $\phi = [\rho, u, v, w, p, k, \omega]^T$ represents any of the primitive variables and the $f_s(\cdot)$ functions represent sine or cosine functions. A Manufactured Solution for the x-component of velocity in a 3D domain is shown in Figure 1(a) and a smooth analytic mass source term in a 3D domain is shown in Figure 1(b). The specific values for the constants in the above equation are given in Appendix B.



(a) (b)

Figure 1 (a) Manufactured solution of u-velocity, and (b) Mass source term

The Manufactured Solutions need to be modified for the verification of the boundary conditions, and this is explained further in Section 5. For verification of the time accuracy of unsteady flows, the Manufactured Solutions will include a time term as well. For example, the Manufactured Solution used for a 2D unsteady problem is of the form shown in Eq.10.

$$\phi(x,y,t) = \phi_0 + \phi_t f_s\left(\frac{a_{\phi t}\pi t}{T}\right) + \phi_x f_s\left(\frac{a_{\phi x}\pi x}{L}\right) + \phi_y f_s\left(\frac{a_{\phi y}\pi y}{L}\right) + \phi_{xy} f_s\left(\frac{a_{\phi xy}\pi xy}{L^2}\right) \tag{10}$$

3.3 Ratio of Source terms

In the process of selecting the Manufactured Solution for code verification purposes, it is required that different terms in the governing equations are roughly the same order of magnitude such that contribution from each term in the governing equation is of same order of magnitude. This prevents the larger magnitude terms from masking errors in other terms of smaller magnitude. In the current work, during the verification of the laminar Navier-Stokes equations, a constant viscosity value of 10 Nm/s^2 is used such that there is an approximately equal contribution from the inviscid and viscous terms. Similarly, during the verification of turbulence models, the Manufactured Solutions are generated such that all the terms in the turbulence models are roughly the same order of magnitude over the domain considered for verification. For example, in the source terms generated from the turbulent kinetic energy equation, the ratios of the convection, diffusion, and production terms to the destruction terms are calculated to check that all the terms in the turbulent kinetic energy equation are of similar orders of magnitude. As an example, the ratio of the production term to the destruction terms in the turbulent kinetic energy equation in a 2D rectangular domain is shown in Figure 2. Similar source term ratios for the turbulent dissipation rate equation are calculated to check that all the terms are of similar orders of magnitude [23].

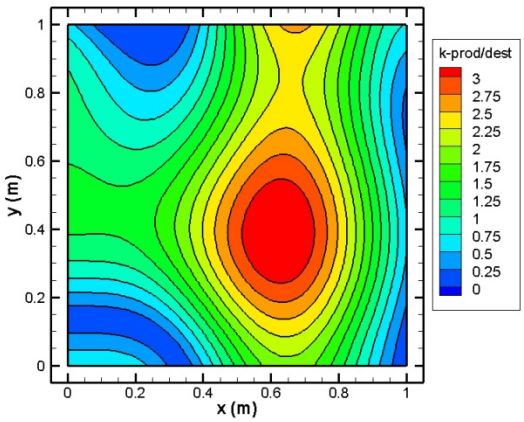


Figure 2 Ratio of the production term to the destruction term in turbulent kinetic energy equation [Ref. 23]

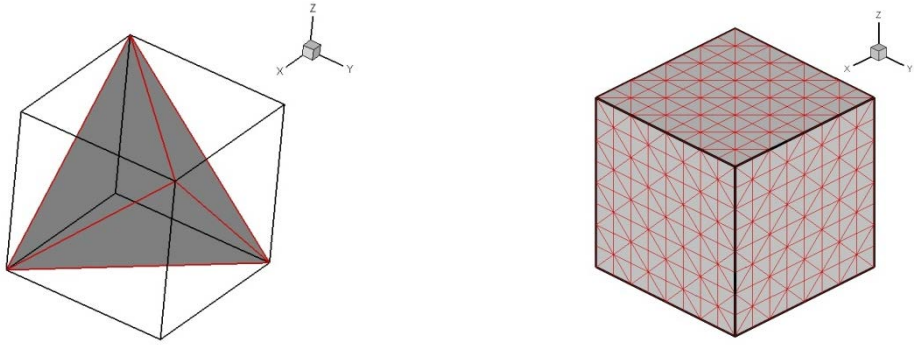
4. Systematic Mesh Refinement and Mesh Generation

The Loci-CHEM finite volume CFD code is verified on different mesh types. In order to verify all mesh transformations are coded correctly, the code is run on the most general mesh types [3, 30] which include meshes with mild skewness, aspect ratio, curvature, and stretching. A 2D hybrid mesh which includes quadrilateral and triangular cells with curvilinear boundaries, skewed cells, and stretched cells is considered as the most general mesh topology for 2D verification. Similarly a 3D hybrid mesh which include tetrahedral cells, prismatic cells, and hexahedral cells with curvilinear boundaries, skewed, and stretched cells is considered as the most general mesh topology for 3D verification.

4.1 Systematic Mesh Refinement

Systematic mesh refinement [3] is defined as uniform and consistent refinement over the spatial domain. A mesh is said to be uniformly refined when the mesh is refined by the same factor over the entire domain and in all the coordinate directions. A mesh is said to be consistently refined if the mesh quality stays constant or improves with mesh refinement. (Note that the order of accuracy test will be easier for the code to pass if the mesh quality improves with refinement.) As discussed in Ref. 3, code order of accuracy verification requires systematic mesh refinement. In the case of structured meshes, coarsening of the meshes for the verification purpose is straightforward. A coarse mesh is generated from a fine mesh by removing every alternate mesh point or mesh line to produce mesh levels with a refinement factor of two. In the process, the mesh quality can be maintained or improved for the structured meshes. But in the case of unstructured meshes refinement/coarsening of meshes with a uniform refinement factor throughout the domain while preserving the mesh quality is more challenging, particularly in 3D. For 2D unstructured meshes, systematic mesh refinement can be achieved by generating an unstructured mesh from a structured mesh by splitting quadrilaterals into triangles using diagonals [30]. To our knowledge, generation of 3D unstructured meshes with uniform refinement while preserving the mesh quality has not yet been achieved using commercial software. Therefore, for code verification purposes, a mesh generation code was developed to generate an unstructured tetrahedral mesh from a 3D structured mesh with hexahedral elements. In the process, a cube will be split into five tetrahedral cells as shown in Figure 3(a). In the case of splitting a cube, the central tetrahedral cell, which does not share a surface boundary with the parent cube, is isotropic and the other four

tetrahedral cells have the same topology with good cell quality. An unstructured mesh with tetrahedral elements generated using the mesh generation code is shown in Figure 3(b). By generating unstructured meshes in this fashion, a uniform and consistent refinement can be achieved by a uniform refinement factor and maintaining the cell quality constant between the mesh levels. Based on this concept of generating an unstructured mesh from a structured mesh, another code for generating 3D hybrid meshes which contain hexahedral, tetrahedral, and prismatic cells with proper connectivity between the different types of cells was also developed. The prismatic cells in the 3D hybrid mesh are generated by diagonally splitting a hexahedral cell into two prismatic cells. To obtain a hybrid mesh from a structured mesh of hexahedral cells, 25 percent of the hexahedral cells are split into prismatic cells, 50 percent of the hexahedral cells are split into tetrahedral cells and other 25 percent are left as hexahedral cells. The 3D hybrid meshes are generated to satisfy the uniform and consistent refinement criteria between the mesh levels (Fig. 4a).

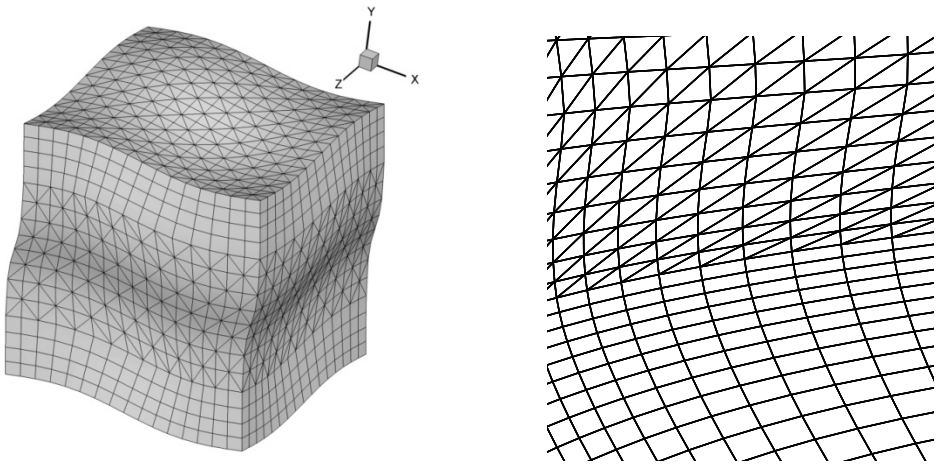


(a) (b)

Figure 3 (a) Hexahedral cell split into five tetrahedral cells, (b) 3D unstructured mesh with tetrahedral cells generated from a Cartesian mesh

4.2 Mesh Topologies

The code is tested on complex hybrid meshes and if the verification tests fail on the complex hybrid meshes, then the code is tested on simpler meshes which isolate the effect of mesh stretching, aspect ratio, skewness, curvature, and cell topology. A 2D hybrid mesh containing quadrilateral cells and triangular cells and a 3D hybrid mesh containing hexahedral cells, tetrahedral cells, and prismatic cells are given in Figure 4. The 3D hybrid mesh has three different layers of cells, each layer consisting of cells of a particular cell topology. In Figure 4(a), the bottom layer consists of hexahedral cells, the top layer consists of prismatic cells, and the center layer consists of tetrahedral cells.

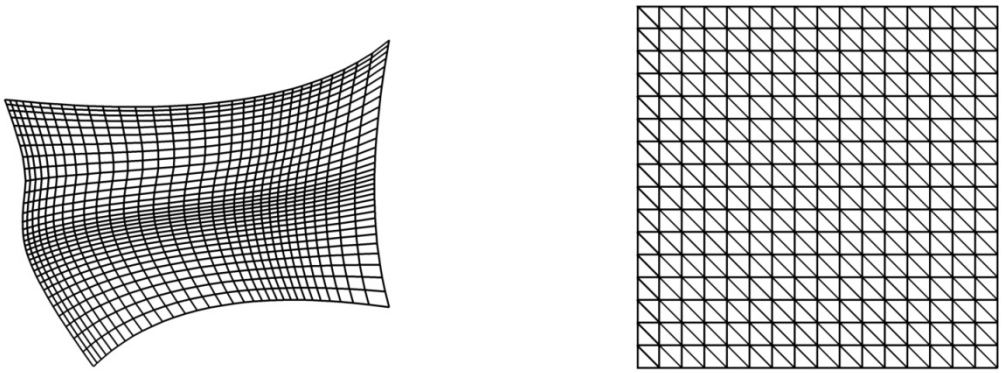


(a) (b)

Figure 4 (a) 3D hybrid mesh, (b) 2D hybrid mesh

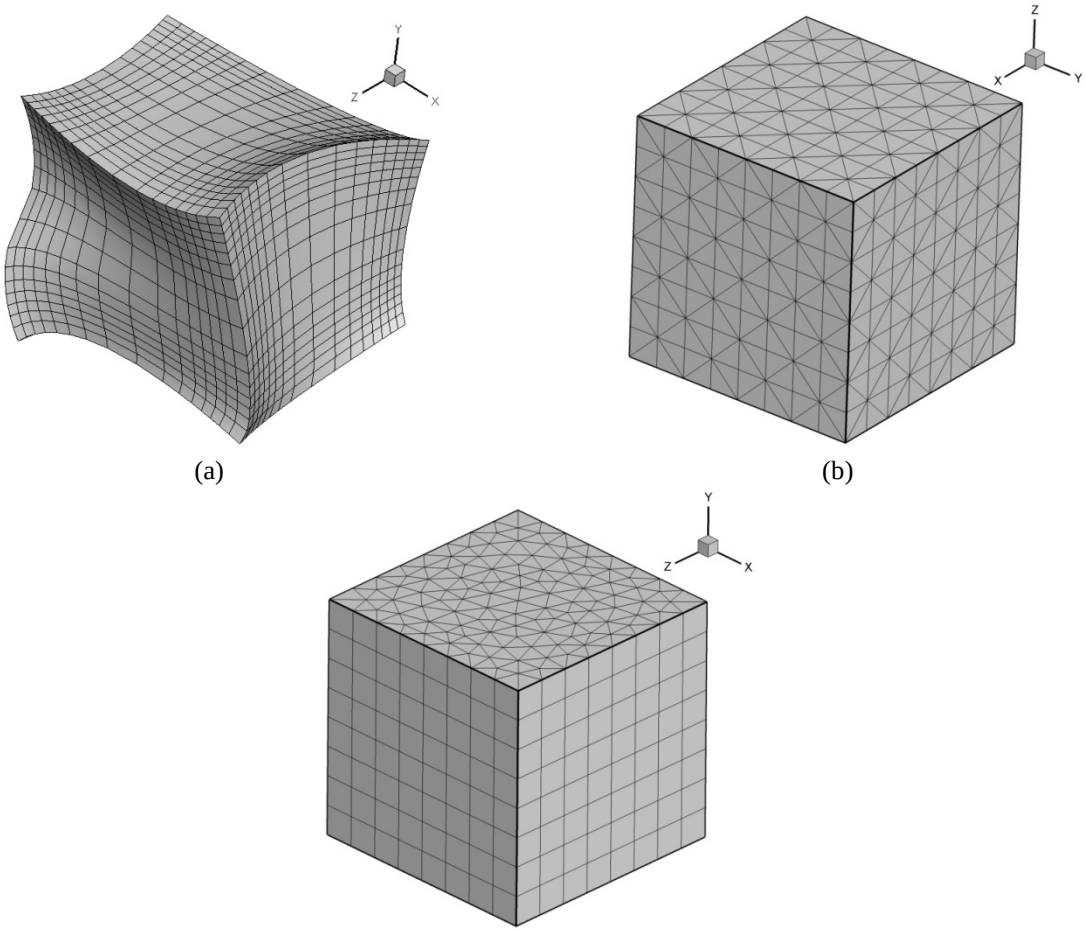
When a verification test fails on a hybrid mesh, testing is then performed on simpler structured and unstructured mesh topologies. In 2D, the simpler topologies include a curvilinear mesh with quadrilateral cells and an unstructured mesh with triangular cells. The mesh with triangular cells is generated by starting with a structured mesh and adding diagonals to quadrilateral cells to make the mesh unstructured. Using this procedure, uniform and consistent mesh refinement can be achieved for unstructured meshes between different mesh levels used for verification purposes. Testing the code separately on the 2D structured and 2D unstructured meshes can find the code sensitivities towards the cell topologies. The 2D skewed curvilinear structured mesh and the 2D unstructured mesh used for testing the finite volume code are shown in Figure 5. The code can also be tested on even simpler meshes [30] like Cartesian, stretched Cartesian, and non-skewed curvilinear meshes to further isolate the effects of mesh stretching, aspect ratio, skewness, and curvature.

Similarly in 3D, when the verification test fails on the 3D hybrid mesh, the code is then tested on meshes with a particular cell type to determine the code sensitivities to the cell topology. Figure 6 shows the 3D curvilinear mesh with highly skewed hexahedral cells, the 3D unstructured mesh with tetrahedral cells, and the 3D unstructured mesh with prismatic cells. The 3D unstructured mesh with tetrahedral cells is obtained by starting from a structured mesh and then splitting each hexahedral cell into five tetrahedral cells. The 3D unstructured mesh with prismatic cells is obtained by starting with a general 2D unstructured irregular mesh and projecting the mesh in the third direction normal to the 2D surface. The code can be tested on even simpler meshes like the 3D Cartesian mesh if required.



(a) (b)

Figure 5 (a) 2D skewed curvilinear mesh with quadrilateral cells and, (b) 2D unstructured mesh with triangular cells



(a) (b)

(c)

Figure 6 (a) Highly skewed 3D curvilinear structured mesh with hexahedral cells, (b) 3D unstructured mesh with tetrahedral cells, and (c) 3D unstructured mesh with prismatic cells

5. Code Verification

Different options tested in the Loci-CHEM CFD code include

- Baseline steady-state governing equations
- Sutherland's law for viscosity
- Thermally perfect thermodynamic model
- Boundary conditions (Adiabatic no-slip wall, Isothermal no-slip wall, Slip wall, Supersonic Inflow, Farfield, Isentropic Inflow, Outflow, Extrapolate)
- Turbulence models (Spalart-Almaras one equation model [32], k-ω turbulence model, k-ε turbulence model)
- Time accuracy for unsteady flows

The options in the finite volume CFD code are verified by comparing the observed order of accuracy of the discretization error calculated for the CFD solutions from multiple systematically-refined meshes to the formal order of accuracy of the numerical method. An option is considered fully verified if it passes this order of accuracy test on the 3D hybrid mesh which has all cell topologies and includes skewness, aspect ratio, curvature, and mesh stretching. During the process of code verification, the code options are tested on different mesh types and the verification results are shown only on 2D and 3D hybrid meshes. When a verification test fails on the hybrid meshes, then the governing equations are tested on other simpler meshes to find whether the discrete formulation of the governing equations is inconsistent on a particular mesh topology or due to the cell quality attributes. Most of the code options in Loci-CHEM are verified, but here we focus on interesting cases that had problems during verification. A summary of the options verified in the finite volume Loci-CHEM CFD code is shown in Appendix A. An option can be assumed verified on a particular mesh even though it is not actually run on that mesh, if that option is already verified on a more complex mesh type.

5.1 Baseline Governing Equations

Verification of the baseline governing equations includes the testing of the Euler equations and the Navier-Stokes equations on the 2D hybrid and 3D hybrid meshes. Successful verification of the Euler equations on a particular mesh and failure of the order of accuracy test for the Navier-Stokes equations means that there is a

problem with the formulation of the diffusion operator on that particular mesh, i.e., an algorithm inconsistency, a coding mistake, or simply mesh quality sensitivity.

The Navier-Stokes equations are verified on the 2D and 3D hybrid meshes and the observed order of accuracy approaches two for the numerical scheme with mesh refinement on both 2D and 3D hybrid meshes. The Navier-Stokes equations are run on six mesh levels of 2D hybrid mesh and four mesh levels of 3D hybrid mesh. The observed order of accuracy is calculated using L_2, L_1 and L_∞ norms of the discretization error. During the code verification process, normally the observed order of accuracy calculated using L_2 and L_1 norms showed similar behavior, but asymptoted to the formal order at a slower rate using L_∞ norm, requiring more mesh levels to see the asymptotic behavior. In this paper, the observed order of accuracy results shown are calculated using only the L_2 norms for clarity, but second order behavior is observed using the L_∞ norms as well. The observed order of accuracy results calculated using the L_2 norm discretization error for the 3D hybrid mesh (Fig. 4a) and the 2D hybrid mesh (Fig. 4b) are shown in Figure 7(a) and Figure 7(b), respectively. In these plots the observed order of accuracy p is on the y-axis and reference cell length h is on the x-axis. The value of h is arbitrarily set to unity on fine grid and hence in these plots the observed order of accuracy approaches two as the h value approaches one. The Euler equations were also verified successfully on the meshes considered in this study, but the results are not shown here.

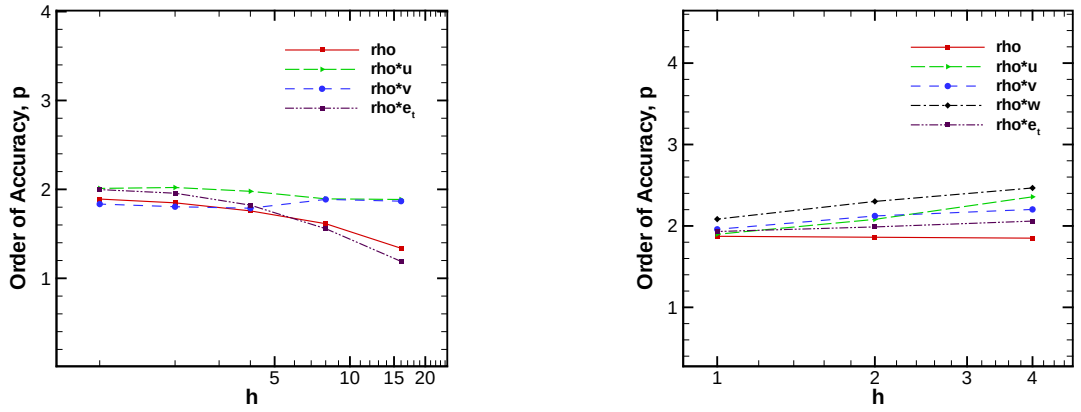
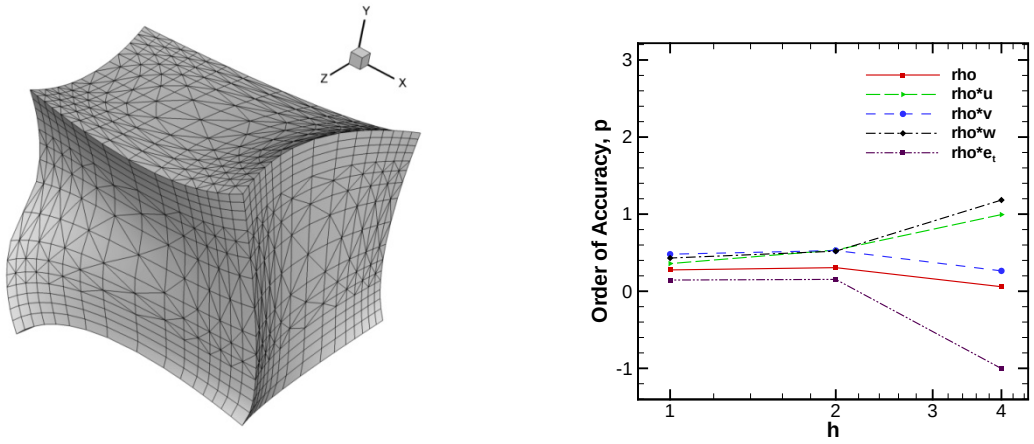


Figure 7 Observed order of accuracy calculated for the Navier-Stokes equations using L2 norm of the discretization error on the (a) 2D hybrid mesh and (b) 3D hybrid mesh

The Navier-Stokes equations were also tested on a highly skewed 3D hybrid mesh shown in Figure 8(a). The Navier-Stokes equations were successfully verified to be second order accurate on the highly skewed 3D curvilinear mesh with hexahedral cells shown in Figure 6(a), but the verification test failed on the highly skewed 3D hybrid

mesh. The observed order of accuracy results are shown in Figure 8(b). From the plot, the observed order of accuracy approaches a value less than one with mesh refinement. The difference between the 3D hybrid mesh (Fig. 4a) and the highly skewed 3D hybrid mesh (Fig. 8a) is only the quality of the cells in the mesh; otherwise both the meshes have the same mesh topology and connectivity. This indicates a problem in the discrete formulation of the governing equations when the cells have a comparatively lower quality. The L_2 norm of the discretization error for both the 3D hybrid mesh and the highly skewed 3D hybrid mesh is compared and it is observed that the error is higher for the skewed hybrid meshes relative to the hybrid meshes for the same number of cells and similar mesh structure. The comparison of the L_2 norm of the discretization error is shown in Figure 9. In the plot, errors shown in the solid lines correspond to the 3D hybrid mesh and the errors shown in dashed line corresponds to the highly skewed 3D hybrid mesh. This plot explains the effect of cell quality on the error in the solution. The error in the solution either decreases slowly or does not decrease with mesh refinement for lower quality meshes.



(a) (b)

Figure 8 (a) Highly skewed 3D hybrid mesh, and (b) observed order of accuracy calculated for the Navier-Stokes equations using L2 norm of the discretization error

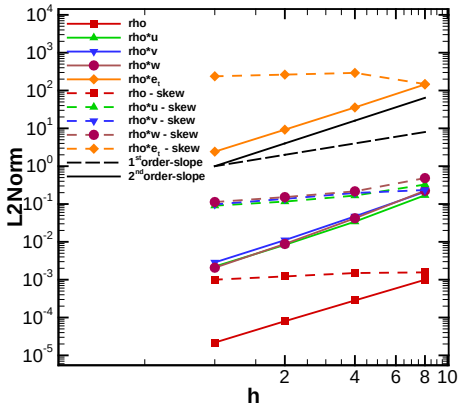
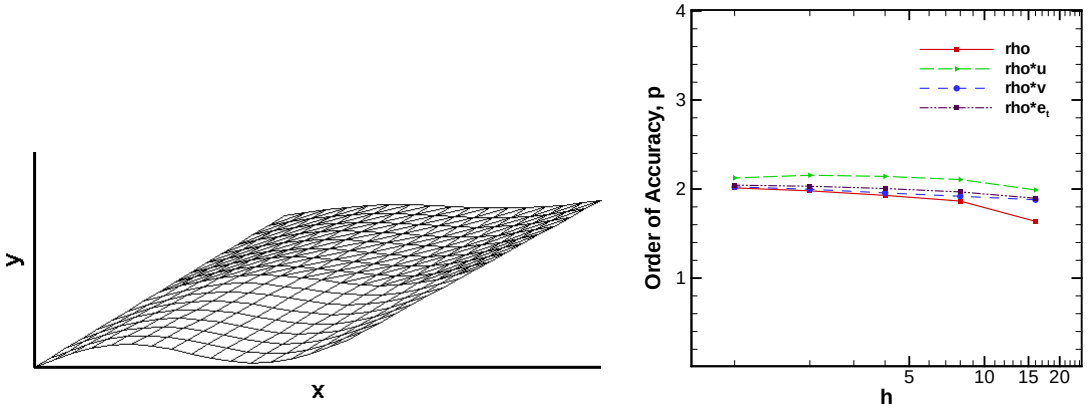


Figure 9 Comparison of discretization error on 3D hybrid mesh (Fig. 4a) and highly skewed 3D hybrid mesh (Fig. 8a)

To further study the discrete formulations of the inviscid and viscous terms in the governing equations, the Euler and Navier-Stokes equations are tested separately on meshes with a particular mesh topology, i.e., either highly skewed tetrahedral cells or highly skewed prismatic cells alone. As an illustration of how the Method of Manufactured Solutions along with the order of accuracy test can be used to find mistakes in the code or inconsistencies in the discrete formulations is explained by testing on different meshes with different cell topologies and different cell quality.

The reason for failure of code verification on the highly skewed 3D hybrid mesh is because of the instability in the inviscid operator. Testing only the heat equation (similar to testing the diffusion operator) on the highly skewed 3D hybrid mesh produced second order accuracy. The Navier-Stokes equations are also tested on the highly skewed 2D hybrid meshes to look at the effect of cell quality of quadrilateral cells and triangular cells on the discrete formulation of the governing equations. A highly skewed 2D hybrid mesh is shown in Figure 10(a). On this mesh, the code is successfully verified and the observed order of accuracy approaches two with mesh refinement which is shown in Figure 10(b). This shows that the finite volume code works fine on highly skewed quadrilateral and triangular cells in 2D.



(a) (b)

Figure 10 (a) Highly skewed 2D hybrid mesh and (b) observed order of accuracy calculated for the Navier-Stokes equations using L2 norm of the discretization error

Other equations verified in the finite volume code are the equation of state, thermally perfect thermodynamic model, and Sutherland's law for viscosity. Verifying the Navier-Stokes equations in the Loci-CHEM code

automatically verifies equation of state. For verifying the transport models, thermal conductivity and viscosity are defined as functions of temperature. They were tested to be second order accurate on 3D hybrid mesh with the observed order of accuracy approaching two with mesh refinement.

5.2 Boundary Conditions

In order to verify the implementation of a boundary condition in a code, the Manufactured Solution can be tailored to exactly satisfy a given boundary condition on a domain boundary. A general approach for tailoring Manufactured Solutions to ensure that a given boundary condition is satisfied along a domain boundary was developed by Bond et al. [6, 24] The approach is explained with a simple example in 2D. The standard form of the Manufactured Solution for the 2D steady-state solution can be written as

$$\phi(x, y) = \phi_0 + \phi_1(x, y) \tag{11}$$

where,

$$\phi_1(x, y) = \phi_x f_s\left(\frac{a_{\phi x}\pi x}{L}\right) + \phi_y f_s\left(\frac{a_{\phi y}\pi y}{L}\right) + \phi_{xy} f_s\left(\frac{a_{\phi xy}\pi xy}{L^2}\right) \tag{12}$$

A boundary in 2D can be represented by a general curve $F(x,y) = C$, where C is a constant. The new Manufactured Solution for verifying boundary conditions can be found by multiplying $\phi_1(x, y)$ term with the function $\left[C - F(x, y)\right]^m$ as shown in Eq. 13.

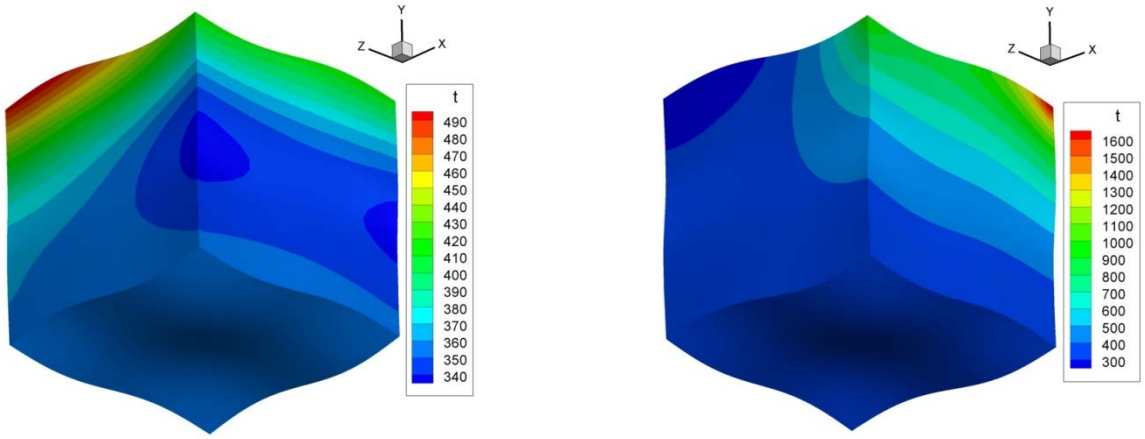
$$\phi_{BC}(x, y) = \phi_0 + \phi_1(x, y)\left[C - F(x, y)\right]^m \tag{13}$$

This procedure will ensure that the Manufactured Solution is equal to constant ϕ_0 satisfying Dirichlet boundary condition along the specified boundary for $m = 1$. For $m = 2$, it ensures that the Manufactured Solution will satisfy both Dirichlet and Neumann (zero gradient) boundary conditions along the specified boundary.

The boundary condition options in the Loci-CHEM code are tested on 2D hybrid meshes and 3D hybrid meshes. In the case of verifying the no-slip wall boundary conditions for the 2D hybrid mesh, a well defined curved boundary on one of the four sides is considered as a no-slip wall and for the 3D hybrid mesh a well defined wavy surface on one side of the domain is considered as no-slip wall boundary. The analytic definition of the curved boundaries considered for verification in both 2D and 3D are defined as shown in Eq. 14.

$$F(x,y) = y - xSin\left(\frac{5\pi}{180}\right) - 0.05Sin(2\pi x) = 0$$
$$F(x,y,z) = 2y - xSin\left(\frac{5\pi}{180}\right) - zSin\left(\frac{6\pi}{180}\right) - 0.06Sin(2\pi x) - 0.05Sin(2\pi z) = 0 \quad (14)$$

These boundaries are used both in the Manufactured Solution and the grid generation. The no-slip wall boundary is tested as an adiabatic boundary and an isothermal boundary. The temperature contours for the case of a no-slip wall defined as an isothermal boundary and an adiabatic boundary in a 3D domain are shown in Figure 11(a) and Figure 11(b), respectively. The bottom wavy surface shown in the figure is the no-slip wall boundary.



(a) (b)

Figure 11 (a) Temperature contours for the isothermal wall for the bottom boundary and (b) temperature contours for the adiabatic wall for the bottom boundary

By testing the no-slip wall as an adiabatic boundary, a Neumann boundary condition for temperature (dT/dn = 0) is verified along with the no-slip condition ($\vec{V}$ = 0) on a particular boundary. The no-slip wall is tested as an adiabatic boundary on both 2D hybrid and 3D hybrid meshes and the observed order of accuracy calculated from the numerical solutions approaches two with mesh refinement. The observed order of accuracy calculated using the L2 norm of the discretization error when the no-slip wall tested as adiabatic boundary on 3D hybrid mesh is shown in Figure 12. Similarly, the no-slip wall is tested as an isothermal boundary, a Dirichlet boundary condition for temperature (T = constant) is verified along with the no-slip condition ($\vec{V}$ = 0) on a particular boundary. The observed order of accuracy for this case approaches two with mesh refinement for both the 2D hybrid and 3D hybrid meshes (not shown). During the verification of no-slip wall boundary condition it was found that the mesh should be normal to the wall. When the mesh was not normal to the wall, the order of accuracy test failed for both 2D and 3D

meshes. This indicated that the numerical formulation is second order accurate only when the mesh is normal to the wall and also the importance of having mesh normal to the wall for achieving the required accuracy.

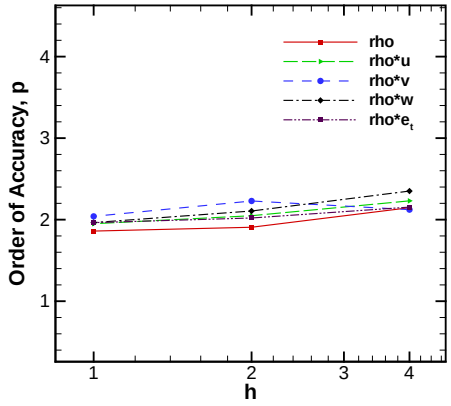


Figure 12 Observed order of accuracy calculated using L2 norm of the discretization error for adiabatic no-slip wall boundary on 3D hybrid mesh

Testing the slip wall boundary is verifying the slip condition $V_n = 0$ on a particular boundary where V_n is the velocity component normal to the surface. Also, on a slip wall boundary, the viscous stress terms need to be zero. The slip wall boundary condition option was tested with both Euler equations and Navier-Stokes equations and found to be second order accurate on 3D hybrid meshes. Other boundary condition options tested in the finite volume code include the farfield boundary condition which is an inflow-outflow characteristic based boundary condition suitable for farfield conditions in external flow for both supersonic and subsonic flow, isentropic boundary condition which preserved total conditions on the boundary, outflow boundary condition which is a characteristic based boundary condition used for both subsonic and supersonic flow, and extrapolated boundary condition which is useful for supersonic outflow conditions. All the above boundary condition options were tested on the 3D hybrid mesh with straight boundaries and were successfully verified to be second order accurate.

5.3 Turbulence Models

Verification of RANS turbulence models provides additional challenges for MMS for different reasons [22]. One of the reasons is that the turbulence models often employ **`min`** or **`max`** functions to switch from one behavior to another, thus causing the source terms to no longer be continuously differentiable. Our approach [22] selects Manufactured Solutions such that they will only activate one branch of the **`min`** and **`max`** functions for a given Manufactured Solution. The turbulence models tested in the finite volume code are the basic k-ω turbulence model

and the k-ε turbulence model which are part of the baseline version of Menter's k-ω model [20] and the Menter's Shear Stress Transport k-ω model [20]. In both the Menter's turbulence models, the k-ω turbulence model gets activated in the boundary layer region and the k-ε turbulence model gets activated away from the wall boundaries in free shear layers. To implement this, the k-ε turbulence model is transformed into a k-ω formulation, and an additional cross diffusion term is added. The k-ω turbulence model and the k-ε turbulence model are blended together using a blending function, F. By setting the blending function F to zero, the cross diffusion term in the turbulent dissipation rate equation is activated and the k-ε turbulence model is tested. By setting the blending function F to unity, the cross diffusion term in the turbulent dissipation rate equation is deactivated and the k-ω turbulence model is tested.

The k-ω turbulence model is tested on the 3D hybrid mesh and the observed order of accuracy approached two with mesh refinement. The observed order of accuracy calculated for the k-ω turbulence model on the 3D hybrid mesh is shown in Figure 13(a). During the testing of k-ε turbulence model, a problem with the turbulent dissipation rate equation was found and the discretization error for that equation did not decrease at the expected rate. The observed order of accuracy of the ρω discretization error norms dropped to zero with mesh refinement, but all the other conserved variable discretization error norms approached two with mesh refinement. See Figure 13(b).

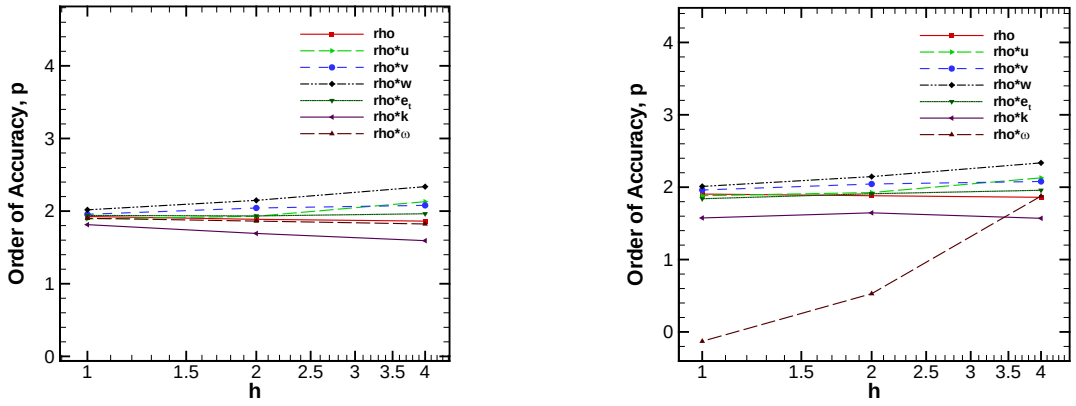
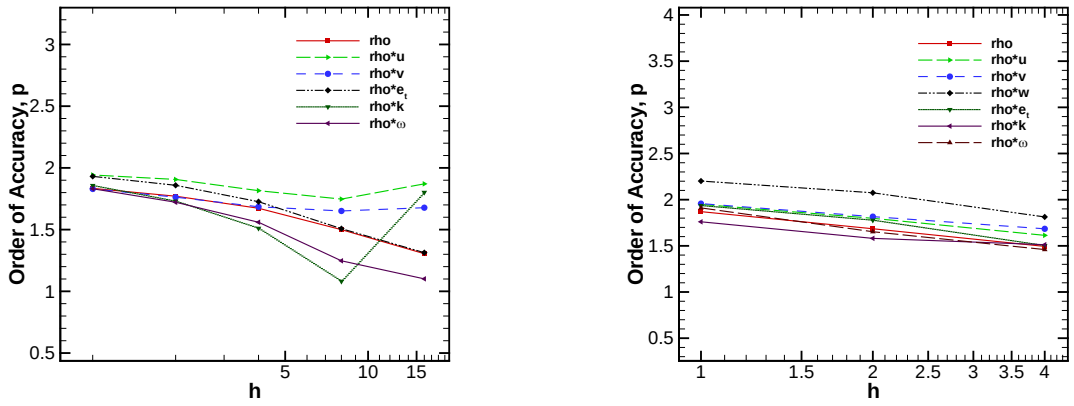


Figure 13 Observed order of accuracy calculated on the 3D hybrid mesh for the (a) the k-ω turbulence model (F = 1) and (b) the k-ε turbulence model (F = 0)

To explore the reason for the failure of the verification test for the k-ε turbulence model on the 3D hybrid mesh, it is tested on simpler meshes. Initially the k-ε turbulence model is tested on the 2D hybrid mesh and it is observed that the verification is successful with all the norms of the discretization errors approaching two with mesh

refinement. The order of accuracy results are shown in Figure 14(a). The above test is also done on a highly skewed 3D curvilinear mesh with hexahedral cells and the k-ε turbulence model is successfully verified with all the norms of the discretization error approaching two with mesh refinement and the calculated observed order of accuracy is shown in Figure 14(b).



(a) (b)

Figure 14 Observed order of accuracy calculated for the k-ε turbulence model (F = 0) on (a) the 2D hybrid mesh and (b) the highly skewed 3D curvilinear (i.e., structured) mesh with hexahedral cells

In addition, the k-ε turbulence model is tested on a 3D unstructured mesh with tetrahedral cells as shown in Figure 3(b) and it is successfully verified with all the norms of the discretization error approaching two with mesh refinement. The observed order of accuracy results are shown in Figure 15. By testing on different meshes, it can be concluded that there is an issue in the discrete formulation of some of the terms in the turbulent dissipation rate equation as it works correctly only on 2D mesh topologies, 3D structured mesh topologies, and 3D unstructured mesh with tetrahedral cells but it fails on 3D unstructured mesh topologies with skewed cells. Performing these tests, the reason for the failure of verification tests for the k-ε turbulence model on the 3D hybrid mesh is isolated to the cross-diffusion term in the turbulent dissipation rate equation and to a particular cell topology.

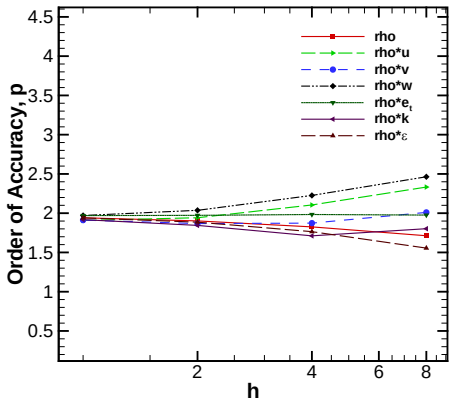


Figure 15 Observed order of accuracy for k-ε turbulence model

The Spalart-Almaras turbulence model has also been verified on 3D unstructured and 3D structured curvilinear meshes using a statistical approach to MMS [10]. In this approach, a single mesh is considered and shrunk down providing a locally refined mesh and this procedure is used to statistically sample the discretization error in different regions of the domain of interest. Using the statistical approach, the Spalart-Almaras turbulence model was successfully verified to be second order accurate (see Ref. 10 for details).

5.4 Time Accuracy of Unsteady Flows

It is more difficult to apply the verification procedure using the order of accuracy test to problems that involve both spatial and temporal discretization, especially when the spatial order of accuracy is different from the temporal order. A combined spatial and temporal order verification method has been developed by Kamm et al. [31] In their approach, they use a Newton-type iterative procedure to solve a coupled, nonlinear set of algebraic equations to calculate the coefficients and observed order of accuracies for the spatial and temporal terms in the discretization error expansion. In this work, we propose a simpler approach for spatial and temporal order verification.

Neglecting the higher order terms, the discretization error for a scheme with spatial and time terms can be written as

$$\varepsilon_{h_x}^{h_t} = g_x h_x^{\hat{p}} + g_t h_t^{\hat{q}} \tag{15}$$

where $\hat{p}$ and $\hat{q}$ are the observed orders of accuracy in space and time, respectively, and g_x and g_t are the coefficients of spatial and time terms, respectively. Similarly, the norm of the discretization error can be found as

$$\left\| \varepsilon_{h_x}^{h_t} \right\| = g_x h_x^{\hat{p}} + g_t h_t^{\hat{q}} \tag{16}$$

Initially, a spatial mesh refinement study is performed with a fixed time step to calculate $\hat{p}$ and g_x using three mesh levels which makes the discretization error equation

$$\left\|\varepsilon_{h_x}^{h_t}\right\| = g_x h_x^{\hat{p}} + \phi \tag{17}$$

where $\phi = g_t h_t^{\hat{q}}$ is the fixed temporal error term. Using three mesh solutions, refined by the factor r, coarse ($r_x^2 h_x$), medium ($r_x h_x$), and fine (h_x), the observed order of accuracy $\hat{p}$ can be calculated (Ref. 3) as

$$\frac{g_x\left(r_x^2 h_x\right)^{\hat{p}} - g_x\left(r_x h_x\right)^{\hat{p}}}{g_x\left(r_x h_x\right)^{\hat{p}} - g_x\left(h_x\right)^{\hat{p}}} = \frac{\left(\left\|\varepsilon_{r_x^2 h_x}^{h_t}\right\| - \phi\right) - \left(\left\|\varepsilon_{r_x h_x}^{h_t}\right\| - \phi\right)}{\left(\left\|\varepsilon_{r_x h_x}^{h_t}\right\| - \phi\right) - \left(\left\|\varepsilon_{h_x}^{h_t}\right\| - \phi\right)}$$

$$\frac{g_x\left(r_x h_x\right)^{\hat{p}}\left[r_x^{\hat{p}} - 1\right]}{g_x\left(h_x\right)^{\hat{p}}\left[r_x^{\hat{p}} - 1\right]} = \frac{\left\|\varepsilon_{r_x^2 h_x}^{h_t}\right\| - \left\|\varepsilon_{r_x h_x}^{h_t}\right\|}{\left\|\varepsilon_{r_x h_x}^{h_t}\right\| - \left\|\varepsilon_{h_x}^{h_t}\right\|}$$

$$\hat{p} = \frac{\ln\left(\dfrac{\left\|\varepsilon_{r_x^2 h_x}^{h_t}\right\| - \left\|\varepsilon_{r_x h_x}^{h_t}\right\|}{\left\|\varepsilon_{r_x h_x}^{h_t}\right\| - \left\|\varepsilon_{h_x}^{h_t}\right\|}\right)}{\ln\left(r_x\right)} \tag{18}$$

where r_x is the spatial refinement factor between two mesh levels and the coefficient of the spatial term g_x can be calculated as

$$g_x = \frac{\left\|\varepsilon_{r_x h_x}^{h_t}\right\| - \left\|\varepsilon_{h_x}^{h_t}\right\|}{h_x^p\left(r_x^p - 1\right)} \tag{19}$$

Similarly, a temporal refinement study is performed on a fixed mesh to calculate $\hat{q}$ and g_t using three temporal discretizations, coarse ($r_t^2 h_t$), medium ($r_t h_t$), and fine (h_t). With all the coefficients calculated, the spatial step size and the temporal step size can be chosen such that the spatial discretization error term has the same order of magnitude as the temporal discretization error term.

Once these two terms are the same order of magnitude, combined spatial and temporal order verification is conducted by choosing the temporal refinement factor such that the temporal error term drops by the same order of magnitude as the spatial error term with refinement, i.e., $r_t = r_x^{\hat{p}/\hat{q}}$. Here r_t is the temporal refinement factor, r_x is the spatial refinement factor, $\hat{p}$ is the spatial order and $\hat{q}$ is the temporal order. In our case, the formal order is 2 in both space and time, i.e., $\hat{p} = \hat{q} = 2$. Using this procedure, the unsteady time term is verified on the 3D hybrid mesh (Fig.

4a) and the 2D hybrid mesh (Fig. 4b) for Navier-Stokes equations. The observed order of accuracy on both the meshes approached two with mesh refinement. The results are shown in Figure 16.

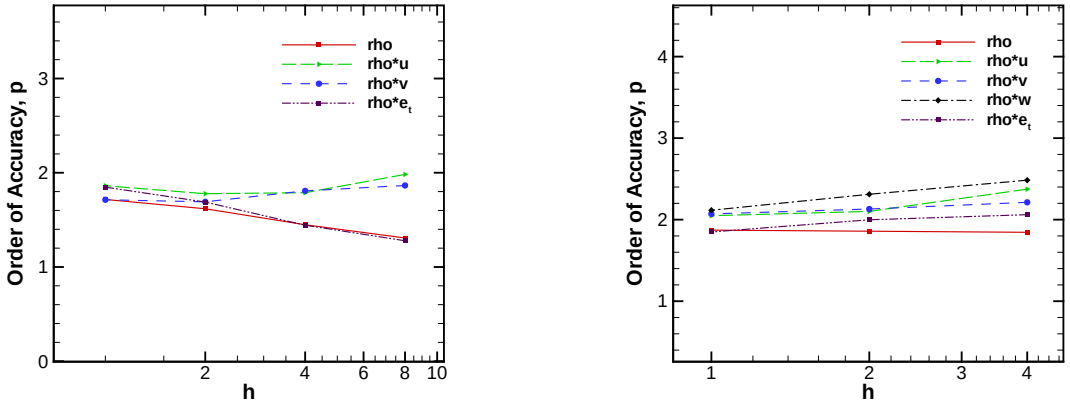


Figure 16 Observed order of accuracy calculated for the unsteady time marching term on (a) the 2D hybrid mesh and (b) the 3D hybrid mesh

6. Issues Uncovered During Code Verification

During code verification studies, some code options failed the order of accuracy test for the first time and the verification studies helped in learning more about the code or the algorithm and ultimately remove some of the mistakes in the code or algorithm. Because the verification procedure is so sensitive to minor issues like the mesh topology or mesh quality, they can often uncover sensitivities to these seemingly minor issues. Issues uncovered during the code verification are documented below.

6.1 Coding Mistakes/Algorithm Inconsistencies

1. A coding mistake was found and corrected in the formulation of the diffusion operator while testing the Navier-Stokes equations in the finite volume code on the 2D skewed curvilinear mesh. Initially, the discretization error did not decrease with mesh refinement and a modification was done to the diffusion operator by Luke [33]. The new diffusion operator rectified the problem and an observed order of accuracy of two was attained with mesh refinement.

2. A coding mistake was found and corrected in the formulation of the diffusion operator while testing the Navier-Stokes equations on 2D unstructured mesh with triangular cells. Initially, the diffusion operator was

found to be only first order accurate on 2D unstructured meshes and after modifying the diffusion operator formulation for unstructured grids, the diffusion operator tested was found to be second order accurate.

3. Testing the k-ε turbulence model on different mesh topologies, it was found that there was an issue in the discrete formulation of the cross-diffusion term in the turbulent dissipation rate equation. The present formulation worked fine on 2D mesh topologies, 3D structured mesh topologies, and 3D unstructured mesh with tetrahedral cells (Figure 15(a)) but failed on 3D unstructured mesh topologies with skewed cells. This issue is under investigation.

6.2 Grid Sensitivities

1. During the verification of the no-slip wall boundary conditions it was found that the mesh should be normal to the wall and the order of accuracy test failed if the mesh was not normal to the wall.

2. Systematic mesh refinement was found to be important for code verification purposes. Failure of code verification tests does not indicate coding errors when meshes are not refined systematically.

3. The discrete formulation of the governing equations was found to be sensitive to highly skewed tetrahedral and prismatic cells. The baseline governing equations were successfully verified on the highly skewed 2D hybrid mesh and highly skewed 3D curvilinear mesh with hexahedral cells, but failed the verification test on a highly skewed 3D hybrid mesh.

7. Conclusions

Comprehensive code verification of an unstructured finite volume code was presented. The Method of Manufactured Solutions was used to generate exact solutions. Different options in the finite volume CFD code were verified which included the baseline governing equations, different boundary condition options, turbulence models, and time accuracy of unsteady flows. All the options were determined to be verified when the observed order of accuracy matched the formal order on the 2D hybrid and 3D hybrid mesh which contained all cell topologies (triangular, quadrilateral, hexahedral, tetrahedral, and prismatic cells). When the verification process failed on any one of these complex hybrid meshes, then simpler meshes were considered to isolate the problem. In addition, a method for generating systematically-refined unstructured meshes for code verification was developed which

involved the coarsening of an underlying structured mesh followed by splitting of the hexahedral elements into tetrahedral or prisms. Coding mistakes, algorithm inconsistencies, and mesh quality sensitivities uncovered during code verification were presented. The requirement of mesh normal to the no-slip wall boundaries to achieve accurate result was also uncovered during code verification. By testing the finite volume code on different meshes, the effect of cell quality and cell topology on the accuracy of the code was assessed. Finally, a new code verification technique was developed and tested for the simultaneous verification of spatial and temporal discretizations which can be applied even when the formal order of accuracy in space and time is not the same.

Acknowledgements

This work is supported by the National Aeronautic and Space Administration's Constellation University Institutes Program (CUIP) with Claudia Meyer and Jeffrey Ryback of NASA Glenn Research Center serving as program managers and Kevin Tucker and Jeffrey West of NASA Marshall Space Flight Center serving as technical monitors.

Appendix A

A summary of the several options verified in the finite volume Loci-CHEM CFD code is shown in Figure A1.

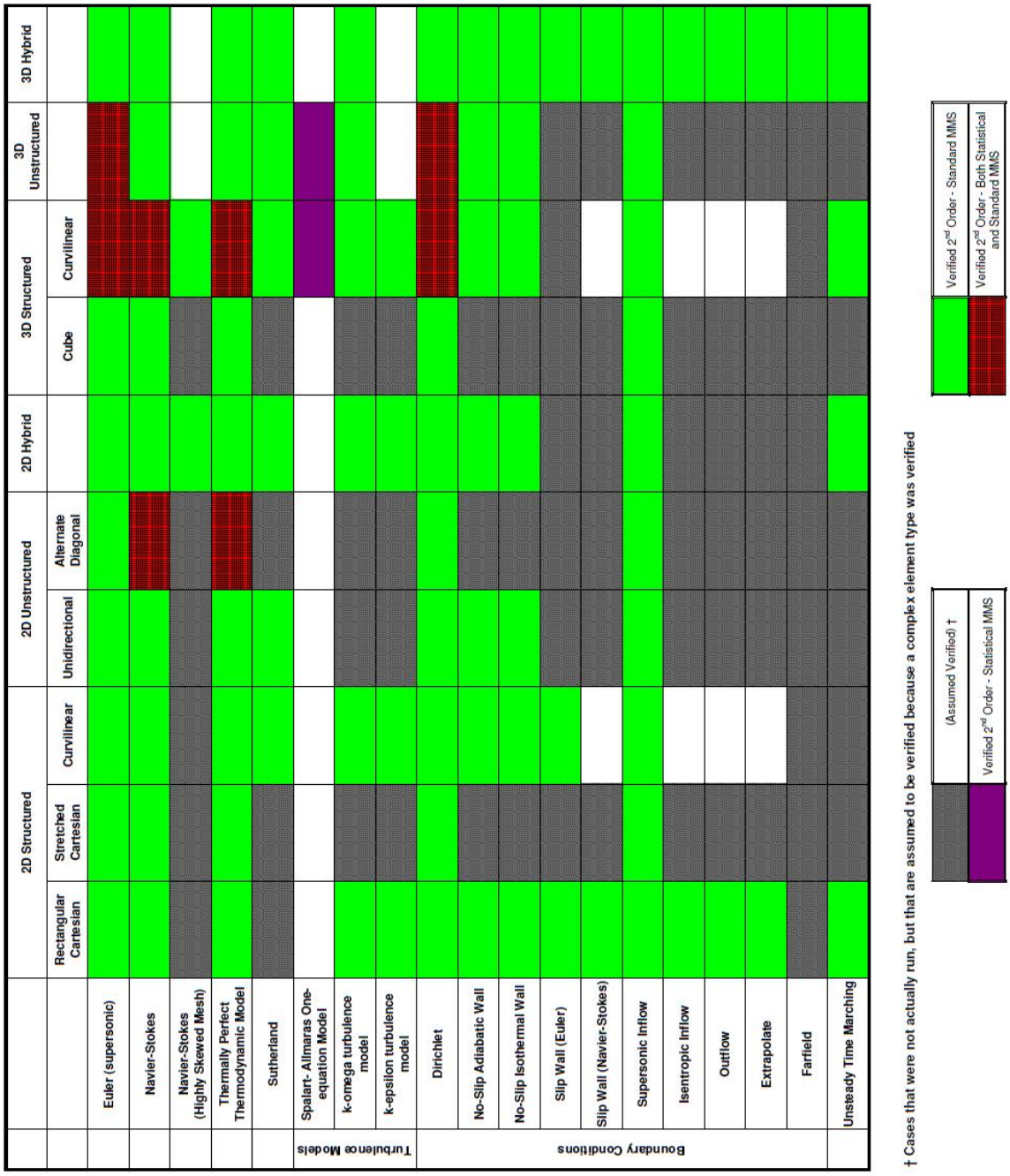


Figure A1. Verification of different options in the finite volume Loci-CHEM CFD code

Appendix B

The constants and the trigonometric functions used in the Manufactured Solution are given in Table B1.

Table B1. Constants for Manufactured Solution

Equation, ϕ	ϕ_0	ϕ_x	ϕ_y	ϕ_z	ϕ_{xy}	ϕ_{yz}	ϕ_{zx}
ρ (kg/m^3)	1	0.15	-0.1	0.1	0.08	0.05	0.12
u (m/s)	70	7	-15	-10	7	4	-4
v (m/s)	90	-5	10	5	-11	-5	5
w (m/s)	80	-10	10	12	-12	11	5
p (N/m^2)	1×10^5	0.2×10^5	0.5×10^5	0.2×10^5	-0.25×10^5	-0.1×10^5	0.1×10^5
k (m^2/s^2)	780	160	-120	80	80	60	-70
ω (1/s)	150	-30	22.5	20	40	-15	25
Equation, ϕ		$a_{\phi x}$	$a_{\phi y}$	$a_{\phi z}$	$a_{\phi xy}$	$a_{\phi yz}$	$a_{\phi zx}$
ρ (kg/m^3)		0.75	0.45	0.8	0.65	0.75	0.5
u (m/s)		0.5	0.85	0.4	0.6	0.8	0.9
v (m/s)		0.8	0.8	0.5	0.9	0.4	0.6
w (m/s)		0.85	0.9	0.5	0.4	0.8	0.75
p (N/m^2)		0.4	0.45	0.85	0.75	0.7	0.8
k (m^2/s^2)		0.65	0.7	0.8	0.8	0.85	0.6
ω (1/s)		0.75	0.875	0.65	0.6	0.75	0.8
Equation, ϕ		f_s (x-term)	f_s (y-term)	f_s (z-term)	f_s (xy-term)	f_s (yz-term)	f_s (zx-term)
ρ (kg/m^3)		cos	sin	sin	cos	sin	cos
u (m/s)		sin	cos	cos	cos	sin	cos
v (m/s)		sin	cos	cos	cos	sin	cos
w (m/s)		cos	sin	cos	sin	sin	cos
p (N/m^2)		cos	cos	sin	cos	sin	cos
k (m^2/s^2)		cos	cos	sin	cos	cos	sin
ω (1/s)		cos	cos	sin	cos	cos	sin

References

[1] Roache PJ. Verification and Validation in Computational Science and Engineering. New Mexico: Hermosa Publishers; 1998.

[2] Roy CJ. Review of Code and Solution Verification Procedures in Computational Simulation, Journal of Computational Physics, Vol. 205, No. 1, pp. 131-156, 2005.

[3] Oberkampf WL, Roy CJ. Verification and Validation in Scientific Computing, Cambridge University Press, Cambridge, 2010.

[4] Roache PJ, Steinberg S. Symbolic Manipulation and Computational Fluid Dynamics, AIAA Journal, Vol. 22, No. 10, pp. 1390-1394, 1984.

[5] Roache PJ. Code Verification by the Method of Manufactured Solutions, Journal of Fluids Engineering, Vol. 124, No. 1, pp. 4-10, 2002.

[6] Knupp P, Salari K. Verification of Computer Codes in Computational Science and Engineering, Rosen, K.H. (ed), Chapman and Hall/CRC, Boca Raton, FL, 2003

[7] Roy CJ, Nelson CC, Smith TM, Ober CC. Verification of Euler / Navier-Stokes Codes using the Method of Manufactured Solutions, International Journal for Numerical Methods in Fluids, Vol. 44, No. 6, pp. 599-620, 2004.

[8] Smith TM, Ober CC, Lorber AA. SIERRA/Premo – A New General Purpose Compressible Flow Simulation Code, AIAA Paper 2002-3292, 2002.

[9] Nelson CC, Power GD. CHSSI Project CFD-7: The NPARC Alliance Flow Simulation System, AIAA Paper 2001-0594, 2001.

[10] Hebert S, Luke E. Honey, I Shrunk the Grids! A New Approach to CFD Verification, AIAA Paper 2005-0685, 2006.

[11] Luke EA, Tong XL, Wu J, Cinnella P. CHEM 2: A Finite-Rate Viscous Chemistry Solver – The User Guide, Tech. Rep. MSSU-COE-ERC-04-07, Mississippi State University, 2004.

[12] Diskin B, Thomas JL. Accuracy Analysis for Mixed-Element Finite Volume Discretization Schemes, National Institute of Aerospace Technical Report TR 2007-8, Hampton, VA, 2007.

[13] Thomas JL, Diskin B, Rumsey CL. Towards Verification of Unstructured-Grid Solvers, AIAA Journal, Vol. 46, No. 12, pp. 3070-3079, 2008.

[14] Pelletier D, Roache PJ. CFD Code Verification and the Method of the Manufactured Solutions, 10th Annual Conference of the CFD Society of Canada, Windsor, Ontario, Canada, 2002.

[15] Pelletier D, Turgeon E, Tremblay D. Verification and Validation of Impinging Round Jet Simulations using an Adaptive FEM, International Journal for Numerical Methods in Fluids, Vol. 44, pp. 737-763, 2004.

[16] Eca L, Hoekstra M. An Introduction to CFD Code Verification Including Eddy-Viscosity Models, European Conference on Computational Fluid Dynamics, ECCOMAS CFD 2006, Wesseling, P., Onate, E., and Periaux, J. (eds.), 2006.

[17] Eca L, Hoekstra M. Verification of Turbulence Models with a Manufactured Solution, European Conference on Computational Fluid Dynamics, ECCOMAS CFD 2006, Wesseling, P., Onate, E., and Periaux, J. (eds.), 2006.

[18] Eca L, Hoekstra M, Hay A, Pelletier D. On the Construction of Manufactured Solutions for One and Two-Equation Eddy-Viscosity Models, International Journal for Numerical Methods in Fluids, Vol. 54, No. 2, pp. 119-154, 2007.

[19] Spalart PR, Allmaras SR. A One-Equation Turbulence Model for Aerodynamic Flows, AIAA Paper 92-0439, 1992.

[20] Menter FR. Two-Equation Eddy-Viscosity Turbulence Models for Engineering Applications, AIAA Journal, Vol. 32, No. 8, pp. 1598-1605, 1994.

[21] Kok JC. Resolving the Dependence on Free-Stream Values for the k-ω Turbulence Model, NLR-TP-00205, 1999.

[22] Roy CJ, Tendean E, Veluri SP, Rifki R, Luke EA, Hebert S. Verification of RANS Turbulence Models the Loci-CHEM using the Method of Manufactured Solutions, AIAA-2007-4203, 2007.

[23] Veluri SP, Roy CJ, Luke EA. Comprehensive Code Verification for an Unstructured Finite Volume CFD Code, AIAA-2010-127, 2010.

[24] Bond RB, Ober CC, Knupp PM, Bova SW. Manufactured Solution for Computational Fluid Dynamics Boundary Condition Verification, AIAA Journal, Vol. 45, No. 9, pp. 2224-2236, 2007.

[25] Luke, E. A., and Cinnella, P., "Numerical Simulations of Mixtures of Fluids Using Upwind Algorithms," Computers and Fluids , Volume 36, December 2007, pp. 1547-1566.

[26] Zhang Y, Luke E. Concurrent Composition Using Loci, Computing in Science and Engineering, Volume 11, Issue 3, pp. 27-35, 2009.

[27] Luke E, George T. Loci: A Rule-Based Framework for Parallel Multidisciplinary Simulation Synthesis, Journal of Functional Programming, Volume 15, Issue 03, pp. 477-502, Cambridge University Press, 2005.

[28] Wilcox DC. Turbulence modeling for CFD, 2nd ed., DCW Industries, La Canada, CA, 1998.

[29] Diskin, B. and Thomas, J.L., Notes on accuracy of finite-volume discretization schemes on irregular grids, Applied Numerical Mathematics, Vol. 60, 2010, pp. 224-226.

[30] Veluri SP, Roy CJ, Hebert S, Luke EA. Verification of the Loci-CHEM CFD Code using the Method of Manufactured Solutions, AIAA-2008-661, 2008.

[31] Kamm JR, Rider WJ, Brock JS. Combined Space and Time Convergence Analyses of a Compressible Flow Algorithm, AIAA-2003-4241, 2003.

[32] Spalart PR, and Allmaras SR. A One-Equation Turbulence Model for Aerodynamic Flows, AIAA-92-0439, 1992.

[33] Luke E. On Robust and Accurate Arbitrary Polytope CFD Solvers, AIAA Paper 2007-3956, 2007