Verification and Validation Studies for the LAVA CFD Solver

Shayan Moini-Yekta*[2], Michael F. Barad†[1], Emre Sozer†[2], Christoph Brehm†[2], Jeffrey A. Housman†[1], and Cetin C. Kiris‡[1],

[1]*NASA Ames Research Center, Moffett Field, CA 94035*

[2]*Science and Technology Corporation, Moffett Field, CA 94035*

The verification and validation of the Launch Ascent and Vehicle Aerodynamics (LAVA) computational fluid dynamics (CFD) solver is presented. A modern strategy for verification and validation is described incorporating verification tests, validation benchmarks, continuous integration and version control methods for automated testing in a collaborative development environment. The purpose of the approach is to integrate the verification and validation process into the development of the solver and improve productivity. This paper uses the Method of Manufactured Solutions (MMS) for the verification of 2D Euler equations, 3D Navier-Stokes equations as well as turbulence models. A method for systematic refinement of unstructured grids is also presented. Verification using inviscid vortex propagation and flow over a flat plate is highlighted. Simulation results using laminar and turbulent flow past a NACA 0012 airfoil and ONERA M6 wing are validated against experimental and numerical data.

I. Introduction

Verification and validation are essential for the development of computational fluid dynamics (CFD) solvers. The definitions accepted by AIAA[1] and ASME[2] for verification and validation address the accuracy of numerical solution (verification) and the physical accuracy of a given model (validation). With the improvements in computational power, CFD has become an integral aspect of the design and analysis process in research and industry. Systematic verification and validation of CFD solvers is required in order to gain confidence in the generated numerical solutions. The overall procedure can be characterized by four steps:[3]

Preparation

In the preparation stage, the geometry, initial conditions and flow solver parameters are defined. In this stage, the test case must be thoroughly researched and any available experimental or numerical data must be collected by conducting a thorough literature survey.

Verification

The process of determining that a model implementation accurately represents the developer's conceptual description of the model and the solution to the model.[1]

Validation

Validation addresses the physical accuracy of a given numerical model. This involves comparing results to trusted experimental or numerical solutions for different classes of problems.

*PhD Candidate, Mechanical and Aeronautical Engineering, University of California, Davis, and AIAA Student Member
†Research Scientist, Applied Modeling and Simulation Branch, NAS Division, MS N258-2 and AIAA Senior Member
‡Branch Chief, Applied Modeling and Simulation Branch, NAS Division, MS N258-2 and AIAA Senior Member

Documentation

The final step is to gather the results and present them in a clear manner for future reference. Thorough documentation will also provide all the necessary information and conditions in order to re-run test cases in the future.

Although the requirements for verification and validation are well known and is now receiving increasing scrutiny,[3] the process remains difficult. Often it is a low priority task that is not systematically performed, or completed only for major releases. The goal of this work is to address some of the issues associated with having the verification and validation procedure incorporated during the development of the solver.

A numerical model can be implemented using a reference book or publication but the complexity of modern CFD solvers means verification is not straightforward. One approach to verification is the Method of Manufactured Solutions (MMS) based on analytically derived solutions that verify correctness and order of accuracy. The analytical functions can be non-physical and designed properly will test each term in a numerical model, thus verifying a correct implementation. The advantage of MMS is that the method is sensitive to even minor errors such as division by an incorrect value or a misplaced sign than can otherwise be easily overlooked.

Even with a verified implementation of a model, validation is required to ensure model accuracy on the problem at hand. While a plethora of test cases have been researched and simulated, it is difficult to find benchmark validation cases that are well understood and documented. With a variety of numerical discretization methods and model variations, it becomes increasingly difficult to confirm a correct implementation. For turbulence models in particular, an internet database has been curated by Rumsey and hosted by NASA Langley.[4,5] The goal of the database is to provide publicly available benchmark turbulent test cases and document numerical solutions with comparisons to experimental data where available.

Although validation procedures traditionally rely on comparisons with experimental data, it is important to note that the procedure does not imply that the experimental data is always correct. As discussed by Roach, experimental uncertainty and unknown bias errors can exist in the experimental data.[6] Validation experiments designed with CFD in mind are also not available for all flow conditions. With this in mind, code-to-code validation is an alternative or supplemental procedure that can improve confidence in a CFD solver. Agreement of observed physical flow features and numerical data with different solver discretization and approaches can be a good indicator of credibility.

Most importantly, completion of the verification and validation procedure is time consuming. In particular, the process can be highly labor intensive in the traditional monolithic infrastructure that revolves around one lead developer. The advantage of the monolithic strategy is that changes to the source code are reviewed and implemented by a single person. As development teams increase in size, however, the monolithic strategy becomes inefficient. Furthermore, maintaining a history and log of file changes and software builds becomes a tedious task for the lead developer. To this end, modern software development tends to favor modular infrastructures that promote collaborative development environments.[7] Tools such as Python, version control, and continuous integration provide an open environment for developers to develop code and automate the time-consuming aspects of verification and validation. Moreover, such an infrastructure allows multiple developers to modify source code and collaborate across cubicles or countries.

The present work provides a pragmatic approach for the verification and validation utilizing software engineering tools and the MMS approach applied to the LAVA CFD solver. LAVA is a research CFD solver developed at NASA Ames Research Center in the Applied Modeling and Simulation Branch. Recent published applications of LAVA include space/time convergence analysis[8] and hybrid grid simulations of the launch environment.[9] This work focuses on three grid strategies in LAVA: block-structured Cartesian, arbitrary unstructured and hybrid variations of Cartesian/unstructured. A modern approach is presented to automate the validation and verification effort with applications to Cartesian, unstructured and hybrid grid paradigms.

The first section provides an overview of the strategy adopted by this work for verification and validation. Next, a description of the software tools including the LAVA solver and software engineering tools is discussed. With the strategy and software tools laid out, preliminary results from the Method of Manufactured Solutions are shown for the verification of field equations and turbulence models. Validation is accomplished through the use of benchmark test cases with experimental and code-to-code comparisons. In the final section, a summary of the results and procedure are discussed.

II. Strategy for Verification and Validation

Verification and validation of codes are typically done on a single version of a code and incrementally redone with major updates of the code. In a fast paced development environment, this often means modifications to the code are done without thorough benchmarking. The goal of this work is to apply development methods of software engineering to automate the verification and validation procedure using a modern framework. The framework consists of version control and continuous integration methods to automate the testing procedure throughout the development process. With these methods, rigorous quality control is maintained throughout development and overall productivity is improved.

The strategy in this work is to develop a test suite that is automatically executed by source code modifications monitored by version control or other trigger sources. Trigger sources include the manually or schedule based (daily, hourly, weekly, etc) initiation of the continuous integration software. Each test case consists of test case interface which sets up the input files, executes the solver and post process the results for a specified sweep of variables. Passing criteria are incorporated into the test case interfaces based on metrics such as coefficient of lift or deviation from previous results to communicate to the continuous integration software and notify developers if potential errors have occurred. Furthermore, with the ability to post process solutions as the test cases complete, the results can be automatically compiled into an updated LaTeX based and versioned documentation of the code. The overall strategy for verification and validation is illustrated by a flow chart in Figure 1. Each element in the flow chart is color coded to indicate the software tool used for the individual tasks.

The test suite consists of fundamental tasks such as compilation of the code and execution of unitary tests to evaluate specific functions to difficult 3D flow simulations. Upon successful completion, additional test cases are evaluated. A sequential or parallel execution of the test suite can be implemented. Experience has indicated that setting up a set of fundamental test cases (e.g. checking out the code or compiling the solver) to execute serially and having subsequent test cases performed in a parallel fashion is optimal. This strategy is efficient at detecting larger errors without initiating multiple test cases. The verification and validation test cases shown in this work are applied to LAVA, but can be extended to any solver. The included test cases highlight fundamental test cases and methods for solution analysis and comparison for a wide range of flow conditions and computational methods.

III. Software Tools

III.A. Version Control

Version control tracks changes to source code and files associated with a software project. With version control, a log of modifications and changes is maintained for all files. Integration of version control software is essential for collaborative development environments. Version control maintains a central repository for source codes, and allows developers to checkout copies of the repository on local machines. A full revision history of source code files are maintained and modifications can be merged into the central repository. Subversion (SVN),[10] an open-source, multi-platform source configuration management tool was chosen here for version control. SVN was chosen due to its simple user interface and ease of configurability. Through the use of SVN, developers can collaborate to modify, retrieve and track changes to source code simultaneously over a network.

The advantage of version control for code development is that each developer has access to the entire version history of each file. With command line options such as *revert*, files can be reverted back to the latest version or modification can be finalized using *commit*. Version control is more commonly used for modern computer science development projects and offers a collaborative environment for the development and distribution of source code. In the context of verification and validation, version control can be coupled with a continuous integration tool to maintain tested versions of the software.

III.B. Continuous Integration

Continuous integration (CI) relieves the developers from the burden of manually running and analyzing test cases. With continuous integration, test cases can be setup and automatically executed based on various triggers and time schedules. To this end, Jenkins CI[11] has been incorporated in LAVA. Jenkins is a CI server that allows the automatic monitoring of source repositories, software builds and execution of test cases.

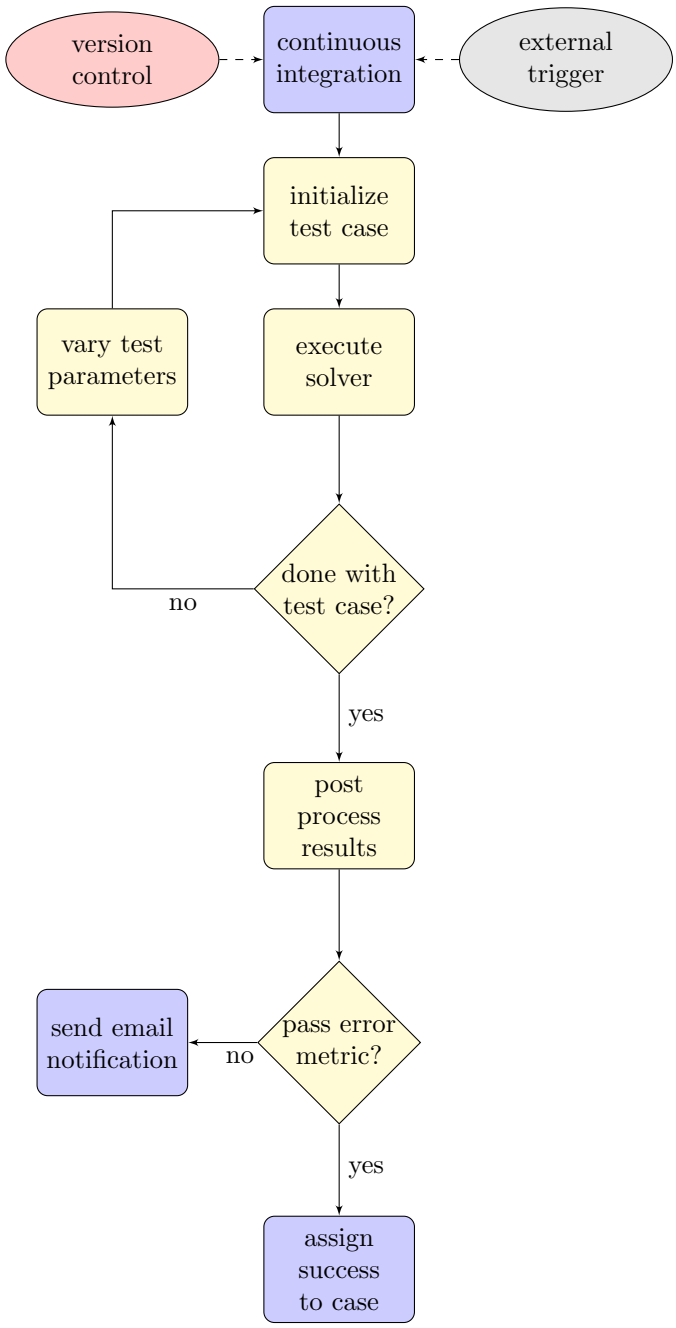


Figure 1. Flow chart illustrating the strategy for verification and validation for a single test case. The color of the elements indicate the software tool used: version control (red), continuous integration (blue), test case interface (yellow) and external inputs (gray).

With over 300 available plugins, Jenkins can incorporate a variety of software including SVN. The approach adopted here is to complete a basic test suite after each source code modification and run larger tests on a weekly or monthly basis. In addition to the automated triggering of test suites, CI can monitor pass and fail criteria and email developers in the event of failure. Continuous integration combined with version control can be used to identify the last valid build for a failed test and assist the developer to pinpoint the specific modification that caused the error.

III.C. Test Case Interface

The strategy adopted in this work wraps each verification and validation test case in the suite in a test case interface script. The main purpose of the interface is to communicate with the continuous integration software and determine whether the test case successfully passed the assigned error metric. A variety of approaches can be used for the development of the interface, but for this work Python was chosen. The advantage of Python is that it is open source software, has a wide variety of available packages and is straight forward to use. Alternatives like Matlab require licenses, and Bash or Perl scripting do not offer the desired functionality. The Python test case interfaces rely on shell commands to create and move between directories, copy or link files and execute the solver. Within each interface, a sweep of variables can be completed (angle of attack, turbulence models, etc) for a test case. With the use of matplotlib, Python can also generate figures to analyze and compare numerical solutions.

An important aspect of the test case interface is determining an appropriate passing error metric. The error metric is case dependent and relies on the available information and post processing methods. Rudimentary metrics can be based on the successful compilation, execution of the solver or known accuracy characteristics. In cases which have analytical solutions, the metric can be based on the order of convergence or percent error tolerances. The metric can also be based on the deviation of the computed solution from previous or available numerical data.

III.D. LAVA

The verification and validation strategy is demonstrated on the LAVA (Launch Ascent and Vehicle Aerodynamics) CFD solver. LAVA is a research code developed in the Applied Modeling and Simulation Branch at NASA Ames Research Center. The LAVA framework is being developed with the emphasis on solving steady and unsteady multi-physics problems. The numerical algorithms are being designed to achieve fast turn-around times. Optimally laying out the data structure based on the underlying mesh being solved on is a key aspect for achieving fast turn-around times. The LAVA solver is highly flexible with respect to the computational mesh. It supports block-structured Cartesian meshes with Adaptive Mesh Refinement (AMR) immersed-boundary capabilities, structured curvilinear overset meshes, unstructured arbitrary polyhedral meshes and hybrid grid coupling.

In order to study physically challenging flow fields such as the launch environment flow, which cover a wide spectrum of relevant physical time and length scales, a large amount of grid points and a long time integration window with relatively small time steps are required. When modeling the near-wall viscous and thermal effects, these problems are solved most economically by utilizing structured curvilinear (overset) meshes which provide a fast accessible memory layout on the computer and the grid point distribution can be directly controlled. Unstructured grids, however, can be generated with significantly turnaround time than structured body-fitted grids. The Cartesian grid solver automatically generates a grid and has a similarly fast accessible memory layout, but cannot economically account for viscous and thermal effect on the walls for high Reynolds number flows.

In LAVA, the unsteady compressible Navier-Stokes equations are primarily solved in finite-volume formulation. On structured grids finite-difference discretizations are also supported. The finite difference discretization provides higher-order (third or fifth) options. A homogeneous mixture model is employed for multi-species simulations where additional $N-1$ species transport equations are added to the Navier-Stokes equations.[12,13] The Spalart-Allmaras (SA)[14] or Shear Stress Transport (SST)[15] turbulence models are supported for RANS/URANS calculations. As a step towards unsteady large eddy simulations, LAVA also supports detached eddy simulation capabilities.[16,17] The dual-time stepping approach is used for implicit time-integration. Standard and strong stability preserving higher-order Runge-Kutta schemes are available for explicit time-integration. Several flux vector splitting and flux difference splitting schemes are available

for the convective terms. The modified Roe scheme described in Housman *et al.*[12,13] and AUSMPW+[18] are available.

In this paper, the hybrid block-structured Cartesian/unstructured grid approach with overset connectivity is utilized. Near-body unstructured meshes resolve viscous boundary layers. An off-body block-structured Cartesian AMR grid is utilized to accurately and efficiently track flow features. The immersed-boundary method is used for regions that do not require viscous spacing. Communication between the off-body and near-body meshes is achieved with overset connectivity and interpolation. Currently, explicit distance-based hole cutting with two layers of fringe points are used. Two-way coupling is done at the sub-iteration level with primitive variables being exchanged between the grid systems.

IV. Verification

IV.A. Method of Manufactured Solutions

To verify the implementation of the governing flow equations, the Method of Manufactured Solutions (MMS) was used. In this method, an analytical solution is specified that does not necessarily satsify the flow equations. To account for this, corresponding source terms are added to the right hand side of the equations. With this approach, the truncation and solution error can be quantified for a numerical scheme. In the absence of an exact solution, MMS enables the verification of the order of convergence of the scheme and verification of the equations and their implementation. Further, manufactured solutions can be designed to mimic complex flow structures similar to actual applications rather than relying on simple fundamental flows. The generation of the manufactured solutions requires special care to ensure a smooth solution and a similar order of magnitude for each term in the governing equations to prevent errors from being obscured. In the present work, only the Euler and Navier Stokes equations are shown using the manufactured solutions presented by Veluri et al.[25] The method has been also applied to the verification of turbulence models and can be extended to boundary conditions. The method is applied to LAVA using both Cartesian and unstructured grids. For all simulations a Roe flux discretization was used with the Cartesian approach while AUSMPW+ was used for the unstructured approach of LAVA.

IV.B. Grid Generation

For the purpose of code verification, systematic mesh refinement is required to assess convergence characteristics of numerical schemes.[26] Systematic mesh refinement is defined as the uniform and consistent refinement over the spatial domain.[25] With structured meshes, the process of grid refinement is straight forward through coarsening and refinement of cells. Refinement can be done by uniform factor-of-two refinement. For unstructured grids, a refinement approach is more challenging. To the knowledge of the authors, current commercial unstructured grid generators do not offer a simple approach to uniform grid refinement. The approach adopted in this work was to develop an algorithm to create the unstructured meshes from a systematically refined structured mesh (quadrilaterals in 2D and hexahedra in 3D). This approach is similar to that of Veluri et al.[25] and provides systematic mesh refinement for both structured and unstructured meshes. Note the grids that are tested are all a single zone with no overset connectivity, however the testing procedure can be applied to hybrid grid methodologies as well.

IV.B.1. 2D Grids

For 2D verification, several variations of grids were generated. Figure 2 illustrates the four variations of the grids. A simple structured Cartesian grid is shown in Figure 2a. Using the Cartesian grid as basis, three unstructured grid variations were generated. First, a uni-directional unstructured diagonal grid (Figure 2b) was produced by dividing structured cells along a single diagonal. Second, the structured cells were divided along alternating diagonals (Figure 2c) for a more complex grid. Finally, a mixed grid was generated by modeling half of the domain with quadrilaterals and the other half with alternating diagonal triangles (Figure 2d). The mixed grid is a closer approximation of the typical grids used with the solver. For all configurations the domain is defined such that $x \in [0, 1]$ and $y \in [0, 1]$.

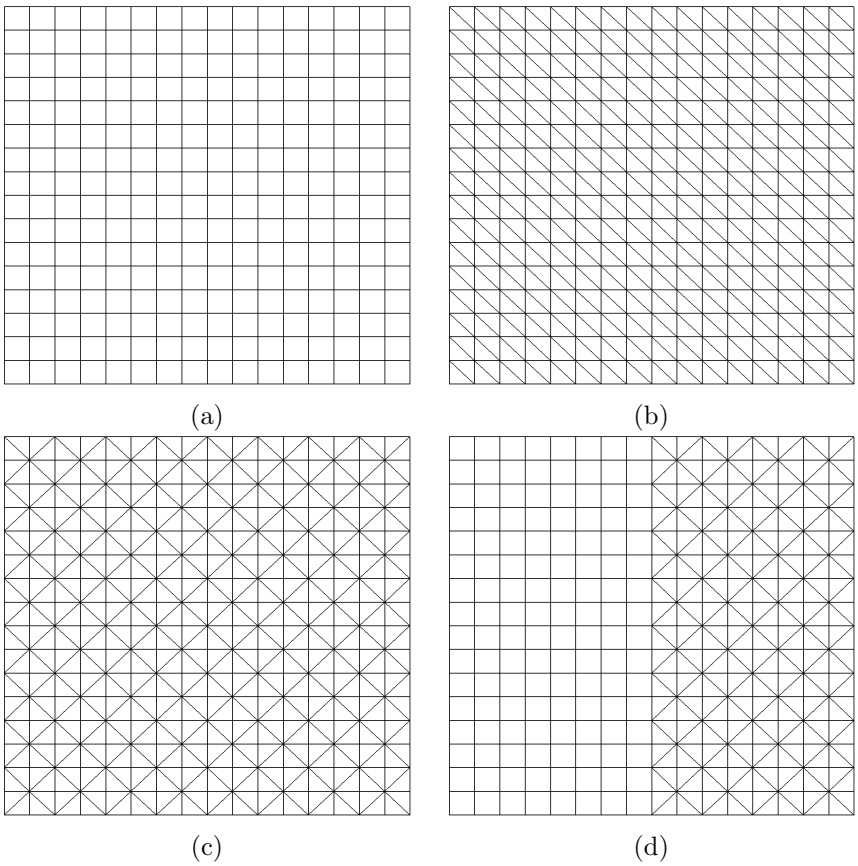


Figure 2. 2D grids used for verification: (a) structured Cartesian, (b) uni-directional unstructured diagonal, (c) alternating unstructured diagonal and (d) mixed unstructured.

IV.B.2. 3D Grids

The 2D grids were extended to 3D for a more comprehensive error study. Figure 3 illustrates the four variations of the grids. Extension of the Cartesian grid to 3D is shown in Figure 3a. A triangular prismatic grid is used as the 3D equivalent of the uni-directional unstructured diagonal grid (Figure 3b). An alternating triangular prismatic grid is also shown in Figure 3c. Finally a mixed grid is generated by applying a 5-tet decomposition on half the domain and an alternating triangular prismatic grid on the second half.

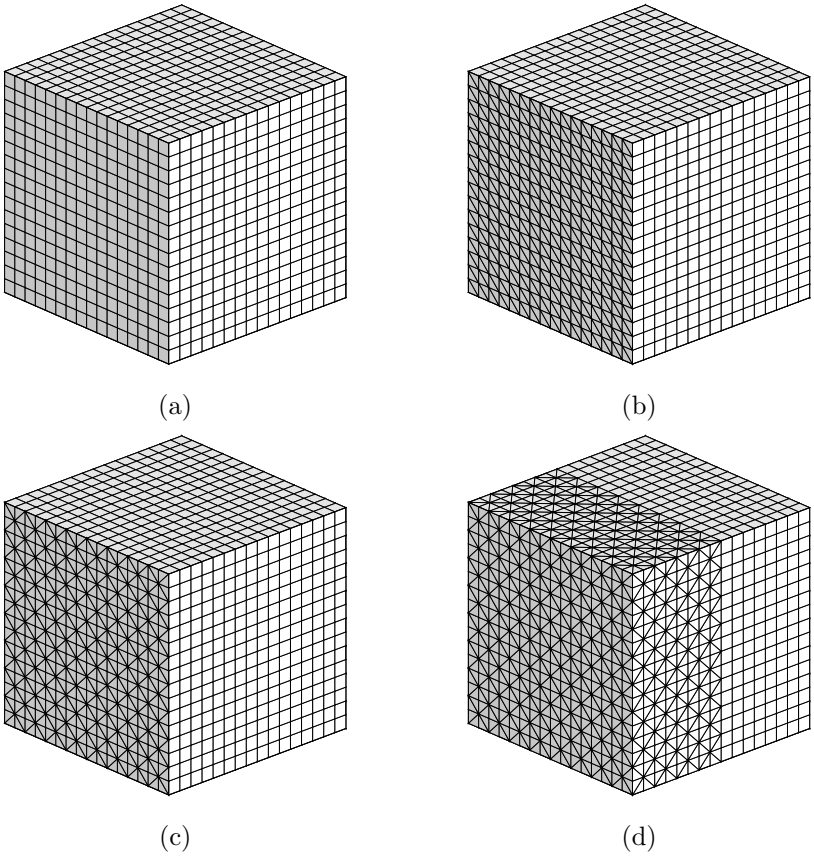


Figure 3. 3D grids used for verification: (a) structured Cartesian, (b) uni-directional unstructured prismatic, (c) alternating unstructured prismatic and (d) mixed tet/prismatic unstructured.

IV.C. 2D Euler Equations

The truncation and solution error rates for the 2D Euler equations are shown in Figures 4-6. Note that all errors are normalized by the error on the finest grid. Each truncation error figure shows the convergence rate for mass conservation (mass), x-momentum (x-mom), y-momentum (y-mom) and the energy equation. For comparison the 1st order, 2nd order and Nth order rates are also shown. Similarly, the solution error indicate the convergence rate for each field variable (pressure, velocity components and temperature) as well as the theoretical convergence rates. The rates are consistent with the expected order of convergence of the schemes used. Matching 2nd order convergence rates are observed for LAVA 2nd order schemes on the Cartesian grid, and similar behavior is observed for higher order scheme truncation errors. For unstructured grids, the convergence rate of the truncation error is expected to be one order less than the discretization (due to the discretization formulation). The unstructured grid also displays more variation in convergence rates for coarser grids due to a smaller grid resolution required for asymptotic behavior. The truncation error is an indication of the formal order of accuracy of a numerical scheme, and measures the rate at which the discretized equations approach the original partial differential equations. The solution error is associated with the observed order of accuracy of a scheme and is more difficult to satisfy.

IV.C.1. Truncation Error

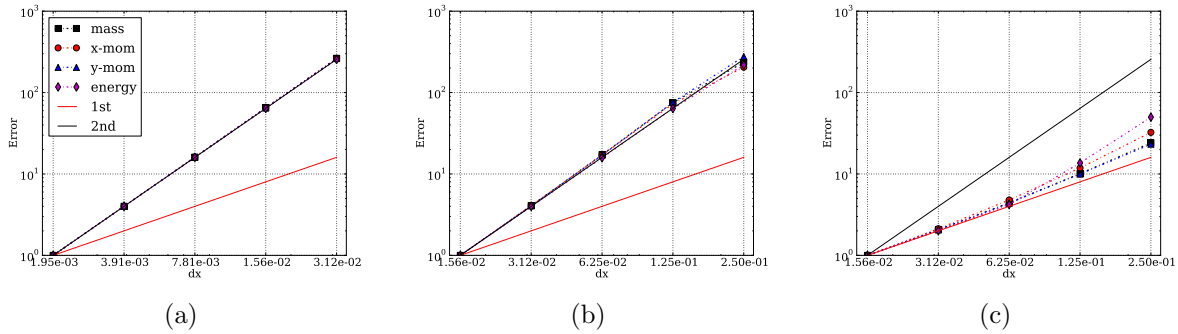


Figure 4. Max-norm of truncation error for 2D Euler equations using LAVA with: (a) Cartesian approach on Cartesian grid, (b) unstructured approach on Cartesian grid and (c) unstructured approach with a mixed unstructured grid.

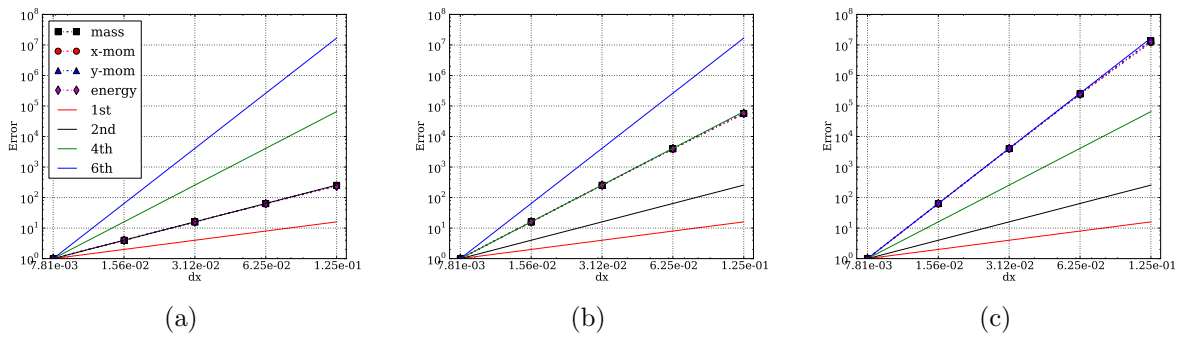
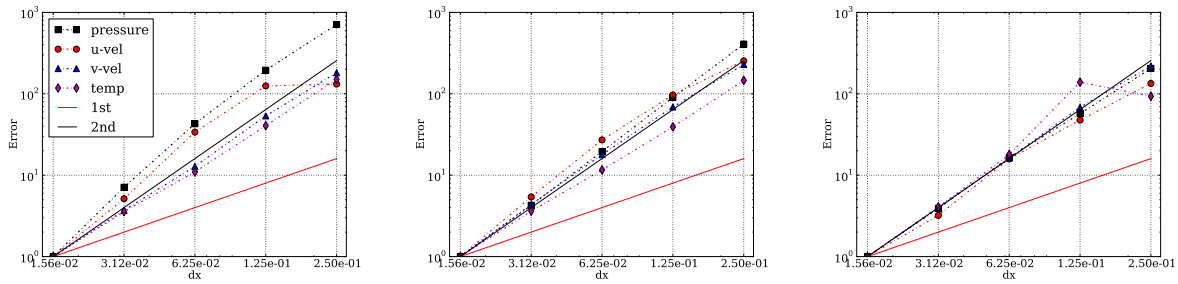


Figure 5. Max-norm of truncation error using the Cartesian approach of LAVA on a Cartesian grid with the 2D Euler equations: (a) 2nd Order, (b) 4th Order, and (c) 6th Order discretizations. Finite difference ("central") discretizations are used without artificial dissipation.

IV.C.2. Solution Error



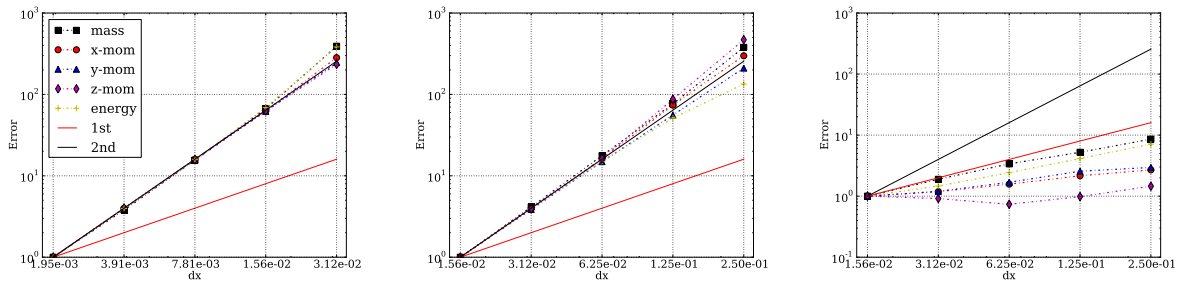
(a) (b) (c)

Figure 6. Max-norm of solution error using the unstructured approach of LAVA for different grids strategies with the 2D Euler equations: (a) Cartesian grid, (b) prismatic grid and (c) mixed grid.

IV.D. 3D Navier-Stokes Equations

The convergence rates for truncation and solution error for the 3D Navier-Stokes equations are shown in Figures 7-8. With second order methods used in both the Cartesian and unstructured approaches of LAVA the expected rate of convergence is two for the truncation and solution errors. Similar trends are observed as in the 2D Euler calculations.

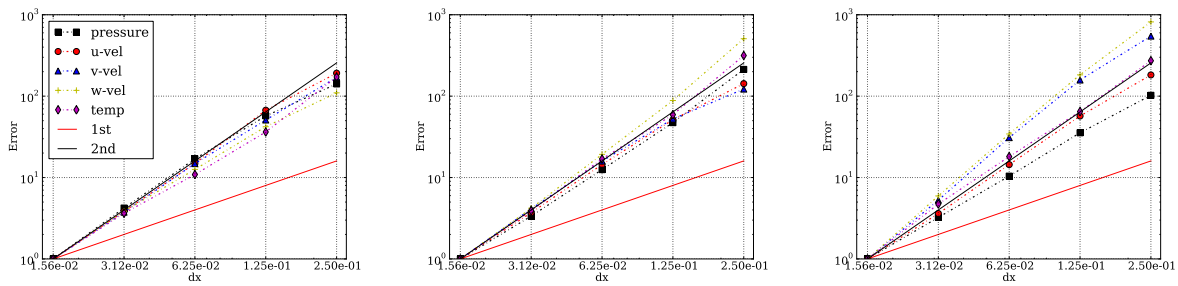
IV.D.1. Truncation Error



(a) (b) (c)

Figure 7. Max-norm of truncation error for 3D Navier-Stokes equations using LAVA with: (a) Cartesian approach on Cartesian grid, (b) unstructured approach on Cartesian grid and (c) unstructured approach on mixed unstructured grid.

IV.D.2. Solution Error



(a) (b) (c)

Figure 8. Max-norm of solution error using unstructured approach of LAVA for different grids strategies with the 3D Navier-Stokes equations: (a) Cartesian grid, (b) prismatic grid and (c) hybrid grid.

IV.E. Laminar Flow Over a Flat Plate

Laminar flow over a flat plate is a well understood and fundamental test case for numerical models. With analytical solutions available and low computational requirements the laminar flat plate is an ideal case for a verification suite. The freestream conditions are: Mach=0.1, T=300K, and Re_x=500.

The domain boundary is given by $x \in [-10, 20]$ and $y \in [0, 20]$ with the leading edge of the flat plate set at the origin and a wall normal grid spacing of $4.607x10^{-3}$. The boundary conditions for this case are characteristic farfield at the inlet and upper domain, slip walls upstream of the plate, no-slip walls on the plate and extrapolation of flow variables and constant pressure at the outlet. For this nearly incompressible, viscous laminar flow conditions the Blasius boundary layer profile can be used for verification of LAVA. In particular the coefficient of friction is given by

$$c_f = \frac{0.664}{\sqrt{Re_x}} \tag{1}$$

while the self-similar velocity profile is governed by

$$2f''' + ff'' = 0, \quad f = f(\eta) \tag{2}$$

The dimensionless similarity variable η is defined as $(U/\nu x)^{1/2}y$. A comparison of the skin friction values is shown in Figure 9 for LAVA and Blasius solutions. The results are all in excellent agreement with the analytical solution. The approximate solution to (2), derived by Ahmad et al.,[27] is given by

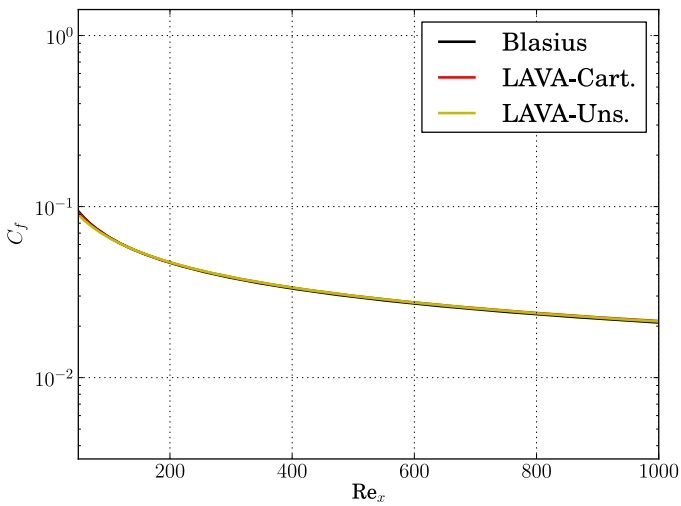


Figure 9. Skin friction coefficient along plate for Blasius solution, and the Cartesian and unstructured approaches of LAVA.

$$f'(\eta) = \frac{0.332057\eta + 0.00059069\eta^4 + 0.00000288\eta^5\exp(0.25\eta^2 - 1)}{1.0 + 0.00869674\eta^3 + 0.00000288\eta^5\exp(0.25\eta^4 - 1.0)} \tag{3}$$

and was used to plot the Blasius solution and is accurate to within a few decimal places of the actual solution. The tangent and normal velocity profiles are compared to the analytical solution at a location near the end of the plate in Figure 10. Again, the results are all consistent with the analytical solutions and illustrate the development of a self similar solution.

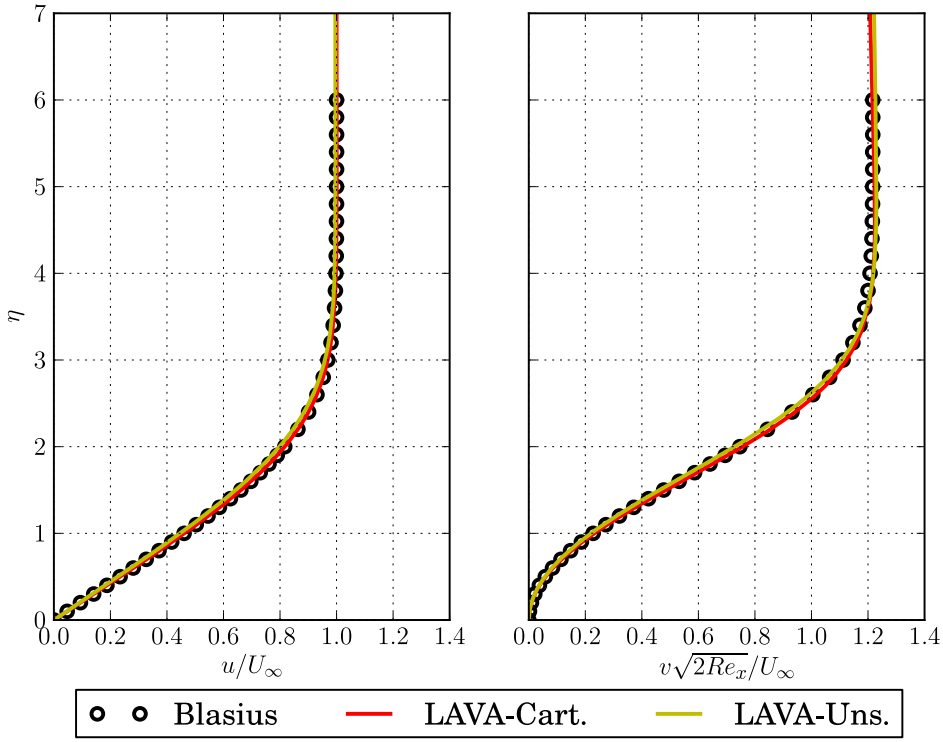


Figure 10. Non-dimensional velocity profiles at the end of the plate for the Blasius solution, and the Cartesian and unstructured approaches of LAVA.

IV.F. Turbulent Flow over a Flat Plate

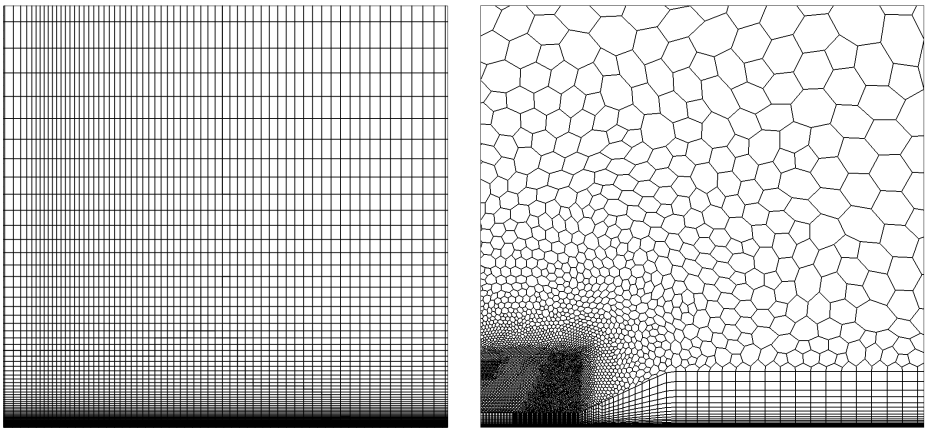
The accuracy of the turbulence models can be assessed using the test case of turbulent flow over a flat plate with zero pressure gradient. This case examines nearly incompressible flow past a flat plate. The free-stream conditions are: Mach=0.2, T=300 K and Re_x=5.00x10^6.

Computational results are compared to those reported by Wieghardt[28] and later included in the 1968 AFOSR-IFP Stanford Conference.[29] Code to code comparison with OVERFLOW was also conducted. OVERFLOW is a well known and validated viscous Reynolds Averaged Navier Stokes (RANS) flow solver that uses structured overset grids.[30] Simulations were completed using AUSMPW+ for the unstructured approach and the Roe flux in the Cartesian approach for LAVA as well as OVERFLOW. Both the SA and SST turbulence models were analyzed.

The computational grid consists of a 5 meter long flat plate at the bottom of a rectangular domain. The domain extends from $x \in [-1, 5]$ and $y \in [0, 1]$ to avoid reflections at the boundaries. Inviscid wall boundary conditions are used in the region before the plate ($x \in [-1, 0]$) and viscous no-slip boundary conditions are applied on the plate ($x \in [0, 5]$). Inlet and outlet boundary conditions are applied at the left and right of the domain (the flow is from left to right) respectively. Characteristic farfield boundary conditions are specified at the upper portion of the domain. Two approaches were taken to create the computational grid for this test problem.

For all grid paradigms, a wall spacing of 3.0×10^{-6} was utilized with a stretching ratio of 1.2 to maintain accuracy. A structured curvilinear grid was used for OVERFLOW and is shown in Figure 11a. The grid features 225 points on the flat plate and 145 points in the normal direction with stretching (1.2 stretching ratio) used to cluster points at the leading edge and in the boundary layer. A fully unstructured polyhedral/prismatic grid was also developed for this test case. A polyhedral and prismatic grid is used to cluster points near the wall and mesh the domain. Additional refinement is placed at the leading edge of the plate to resolve the high gradients. The grid is shown in Figure 11b.

As a first comparison, the skin friction was computed on the flat plate and compared between LAVA



(a) Structured Mesh (b) Unstructured Mesh

Figure 11. Computational grids for flat plate test case using: (a) structured curvilinear and (b) unstructured grid approaches. A subset of the grid at the leading edge of the plate is shown with the bounds $x \in [0, 0.025]$ and $y \in [0, 0.025]$

and OVERFLOW using both SA and SST turbulence models. The results are shown in Figure 12 for a range of Reynolds numbers on the plate surface for the SA turbulence model (similar results are seen with SST). Good agreement is found between the different flow solvers. Note that the SA model consistently generates a slightly larger skin friction coefficient than the SST model independent of solver used. This is likely associated with differences in the turbulence model formulations which can effect the initial eddy viscosity build up at the leading edge of the plate.

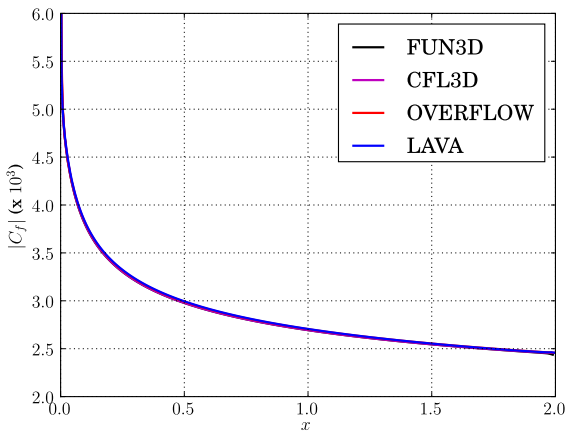
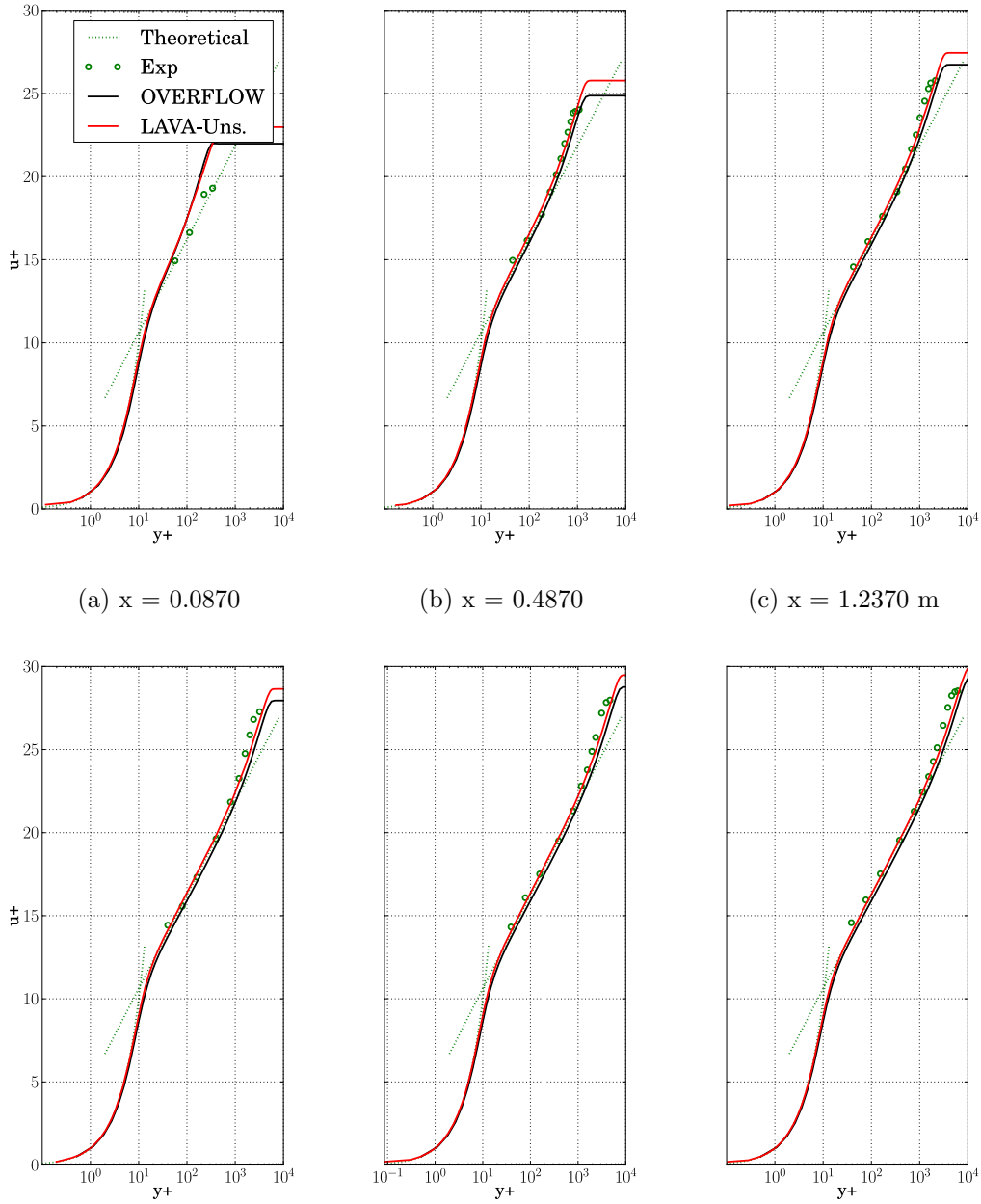


Figure 12. Coefficient of friction distribution on surface of flat plate. Solutions using LAVA, OVERFLOW, CFL3D and FUN3D are shown with SA turbulence model all on the same structured grid.[4,5]

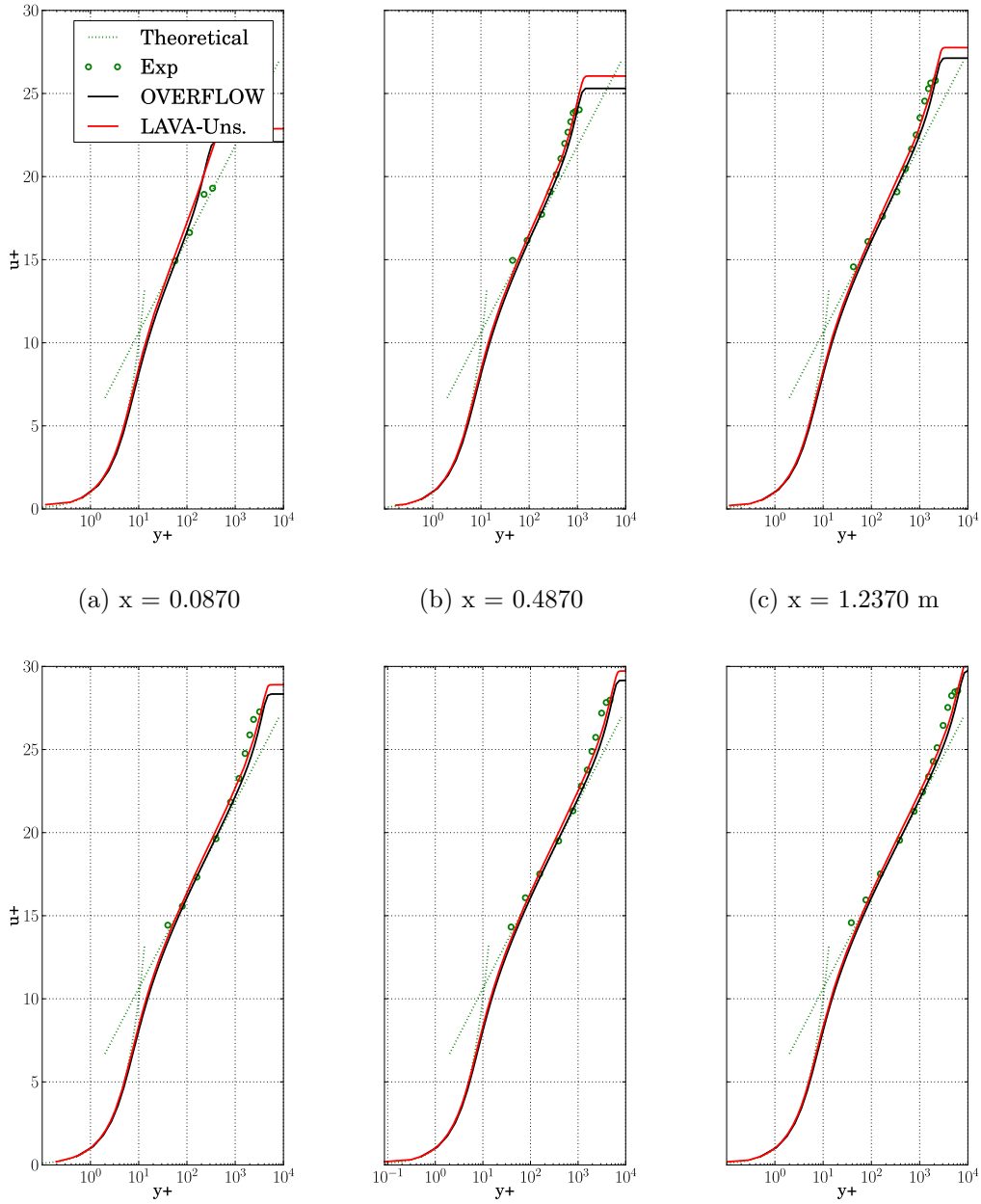
The turbulent flat plate is also a benchmark problem and commonly used to verify and tune turbulence models. In particular, the boundary layer profile is compared with theoretical and experimental data in terms of u^+ and y^+. The boundary layer profiles are shown in Figure 13 and 14 for the SA and SST models respectively. Six stations along the plate are analyzed at $x = (0.08070, 0.4870, 1.2370, 2.2870, 3.4870, 4.6870)$ which correspond to experimental sampling points. Theoretical sublayer and loglayer profiles are also included as reference in the figures as green dotted lines (a Von Karman constant of κ=0.41 is used). As the flow develops, the boundary layer thickens along the plate, which can be seen in the numerical and experimental results. Both LAVA and OVERFLOW are well correlated with the experimental data and theoretical profiles. The LAVA results show a slightly thicker boundary layer profile which can be associated differences in the numerical schemes at the leading edge singularity. Using the effective plate position for comparison plots is a method to eliminate the offset.



(a) x = 0.0870 (b) x = 0.4870 (c) x = 1.2370 m

(d) x = 2.2870 (e) x = 3.4870 (f) x = 4.6870 m

Figure 13. Turbulent boundary layer profiles using the SA turbulence model at multiple stations: (a) x = 0.08070, (b) 0.4870, (c) 1.2370, (d) 2.2870, (e) 3.4870 and (f) 4.6870 m.



(a) x = 0.0870 (b) x = 0.4870 (c) x = 1.2370 m

(d) x = 2.2870 (e) x = 3.4870 (f) x = 4.6870 m

Figure 14. Turbulent boundary layer profiles using the SST turbulence model at multiple stations: (a) x = 0.08070, (b) 0.4870, (c) 1.2370, (d) 2.2870, (e) 3.4870 and (f) 4.6870 m.

IV.G. Inviscid Vortex Propagation

An isolated inviscid vortex propagation test case was chosen to examine the time integration methods of LAVA. This case examines the 2D vortex propagation using the Euler equations with time accurate integration techniques. Furthermore, the propagation behavior of the vortex is highly sensitive to temporal and spatial discretization errors. The non-dimensional initial conditions were chosen to match those published by Pulliam.[32] The free-stream conditions are given by: Mach=0.5, Temp=1, Pressure= = 1.0 and ρ=1. The initial conditions for the isolated vortex are given by

$$\begin{aligned} T &= T_\infty - \frac{V_s^2(\gamma-1)}{16G_s\gamma\pi^2}e^{2G_s(1-r^2)} \\ \rho &= T^{\frac{1}{(\gamma-1)}} \\ u &= M_\infty - \frac{V_s}{2\pi}(y-y_0)e^{G_s(1-r^2)} \\ v &= \frac{V_s}{2\pi}(x-x_0)e^{G_s(1-r^2)} \end{aligned} \tag{4}$$

where the vortex strength is $V_s = 5.0$ and the Gaussian width scale is $G_s = 0.5$. The vortex core is initially set at $x_0 = 5$ and $y_0 = 5$ with vortex core diameter size $C_d = 1.0$ unit and $r = \sqrt{(x-x_0)^2 + (y-y_0)^2}$. Figure 15 shows the density distribution and contours for the initial condition. For the LAVA solvers, a Van Leer flux splitting scheme was used with no gradient limiting and second order dual time stepping schemes. The Cartesian approach of LAVA uses a red-black Gauss-Siedel right hand side discretization while the unstructured approach employs a GMRES solver.

A rectangular computational domain was used with Dirichlet conditions for the exterior boundaries. A refined grid region was used in the region $x \in [0, 120]$ and $y \in [0, 10]$ within a larger coarser domain to minimize boundary condition reflections. Propagation distances of $60C_d$ are used for this analysis which is encompased by the refined grid region. A multi-level resolution Cartesian grid was developed for the computation grid and features a refined interior region embeded within a coarse outer domain. An unstructured tetrahedral grid was also tested with a refined core region that is coarsened out at a stretching ratio of 1.2 to the boundaries. A hybrid grid consisting of an unstructured grid embeded into the Cartesian grid inside half the domain of the refinment region was also examined. With this configuration half of the vortex lies in the Cartesian domain while the other half is within the unstructured domain. The grids are illustrated in Figure 16 for all three gridding paradigms.

Due to the sensitivity of the vortex to numerical errors, a variety of spatial and temporal spacings were considered. The run matrix is summarized in Table 1 where the CFL is defined as $CFL = \frac{\Delta x U_\infty}{\Delta t}$ and freestream conditions are used where Δt is the physical time step. Three different grid resolutions were tested with three Δt steps corresponding to CFL=0.5, 1.0 and 2.0. To assess the temporal convergences four different subiteration values were used $nsub = 10, 20, 50, 100$ for each CFL/grid combination.

To assess the spatial and temporal convergence of the solvers two aspects were analyzed: dissipation and distortion of the vortex core. In order to quantify these two properties the density distribution for the vortex was compared on the full domain and two-dimensional density line extracts along the center of the domain. Temporal convergence was determined by comparison of solution for varying subiteration values on a single

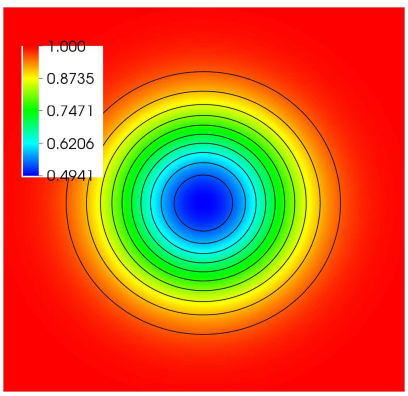


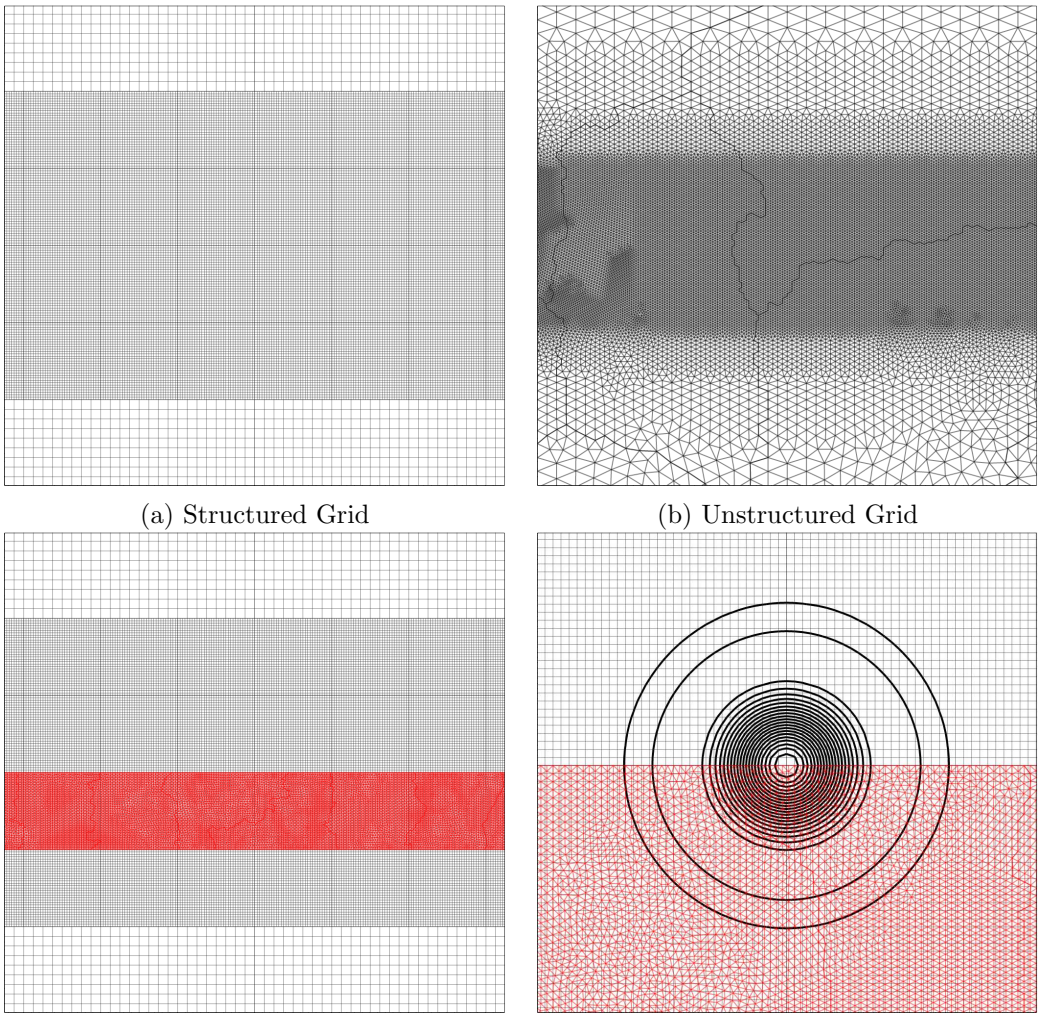
Figure 15. Density distribution for initial condition for inviscid vortex propagation test case.

GRID	$dx = 1.5625e-1$	$dx = 7.8125e-2$	$dx = 3.90625e-2$
CFL=0.5	nsub=10/20/50/100	nsub=10/20/50/100	nsub=10/20/50/100
CFL=1.0	nsub=10/20/50/100	nsub=10/20/50/100	nsub=10/20/50/100
CFL=2.0	nsub=10/20/50/100	nsub=10/20/50/100	nsub=10/20/50/100

Table 1. Run matrix for inviscid vortex propagation test case.

grid. Similarly, spatial convergence was established by comparing the subiteration converged solutions on the three grid levels. Insufficient subiteration convergence can result in linearization errors being accumulated through each time step. Spatial resolution is necessary to resolve the steep gradients in the vortex core and controls the amount of dissipation with the discretized equations.

To assess the spatial and temporal convergence samples were taken along the center of the domain to track the flow quantities in the vortex core. In particular, line samples were computed from the Cartesian, unstructured and hybrid computation grid solutions. Grid convergence is illustrated in Figure 17 by plotting the density distribution along the center of the vortex for all three grid resolutions. For each simulation a CFL=1 and NSUB=50 were chosen to ensure temporal convergence is attained (a minimum of six order of convergence is observed at each time step). The results indicate the coarsest grid is insufficient to resolve



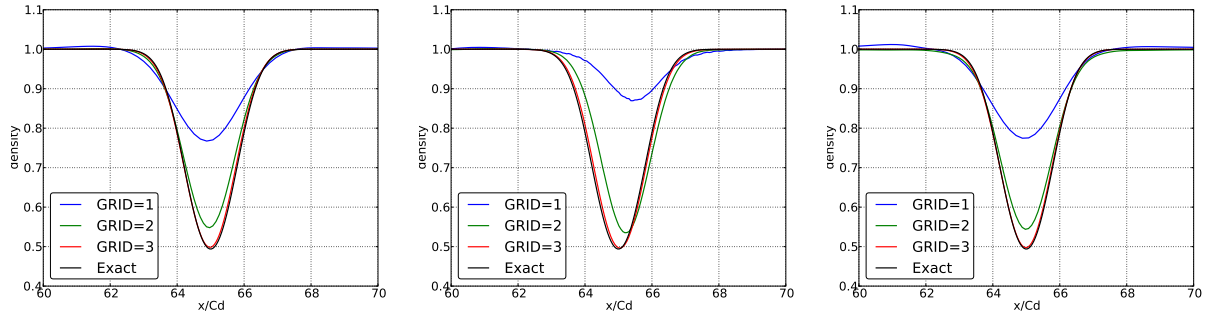
(a) Structured Grid (b) Unstructured Grid

(c) Hybrid Grid (d) Hybrid Grid Closeup

Figure 16. Computational grids for the inviscid vortex propogation test case using: (a) structured Cartesian, (b) unstructured and (c-d) hybrid grid approaches.

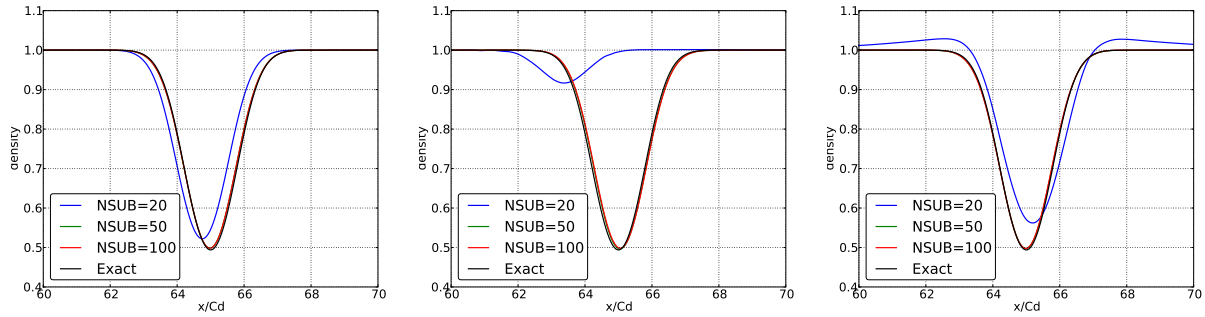
the vortex and results in significant dissipation. For coarse resolutions the vortex core drifts from the center line and results in visibly higher density at the centerline sample. However, the second and third grid results appear similar for all three grid strategies (they are overlapping in the plots). A similar approach was taken to quantify the sub-iteration convergence with the results shown in Figure 18 using CFL=1 and the finest grid resolution. Again, the results show that 20 sub-iterations is insufficient to converge the non-linear system while 50 amd 100 are enough (this is confirmed by viewing residual sub-iteration convergence plots). With insufficient sub-iteration convergence the magnitude and phase of the vortex core are incorrectly predicted.

To determine the correct CFL number, which dictates the numerical time step, the density line plots were compared using 50 sub-iterations and the finest grid resolution. Figure 19 indicates that CFL=2 has noticeable phase error while 0.5 and 1.0 converge to similar predictions. This is consistent with expectations,



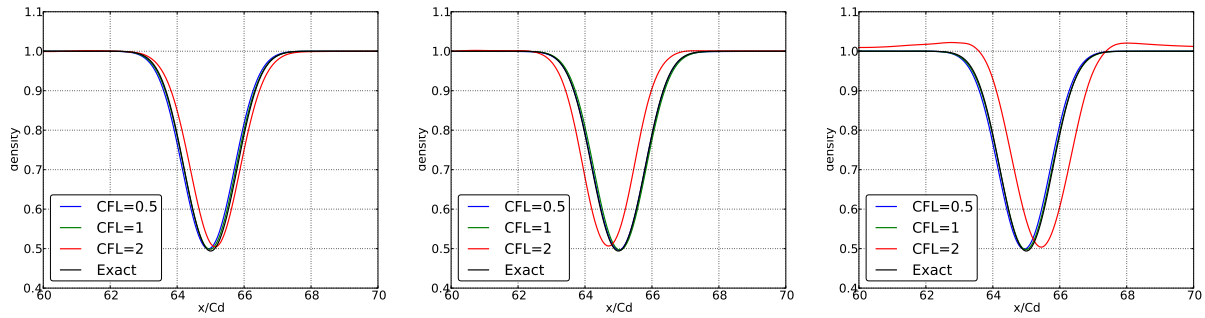
(a) Cartesian Grid (b) Unstructured Grid (c) Hybrid Grid

Figure 17. Grid convergence of vortex core density distribution after $60C_D$ using: (a) structured Cartesian, (b) unstructured and (c) hybrid grid approaches. All simulations are completed with CFL=1 and NSUB=50.



(a) Cartesian Grid (b) Unstructured Grid (c) Hybrid Grid

Figure 18. Sub-iteration convergence of vortex core density distribution after $60C_D$ using: (a) structured Cartesian, (b) unstructured and (c) hybrid grid approaches. All simulations are completed with CFL=1 on the finest grid resolution. Note that NSUB=50 and NSUB=100 are overlapping on the Cartesian grid and the Hybrid grid.



(a) Cartesian Grid (b) Unstructured Grid (c) Hybrid Grid

Figure 19. Time convergence of vortex core density distribution after $60C_D$ using: (a) structured Cartesian, (b) unstructured and (c) hybrid grid approaches. All simulations are completed with NSUB=50 on the finest grid resolution.

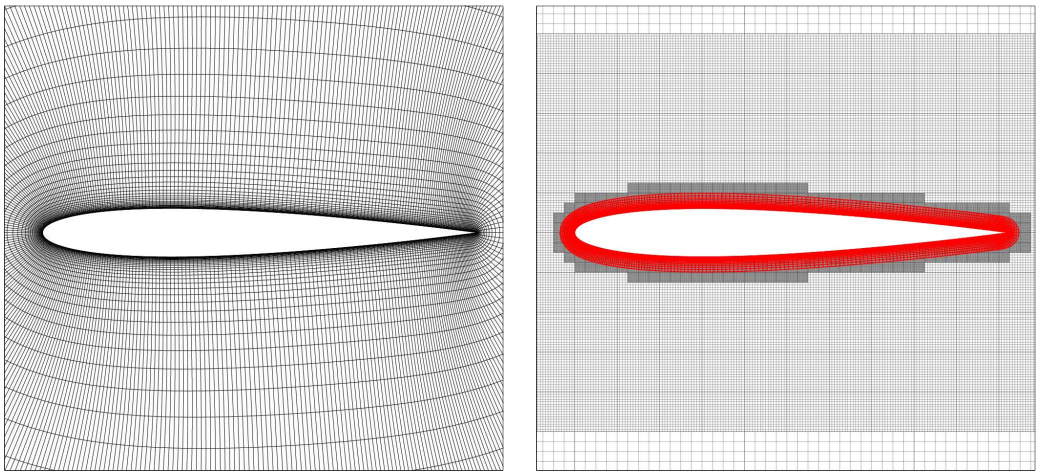
as a large time step has an inherently larger temporal error. Another important aspect of this problem is to illustrate the strength of the hybrid approach of LAVA. Superimposing the density distributions for LAVA indicates that hybrid results in a solution between that of purely Cartesian and unstructured approaches.

V. Validation

V.A. NACA 0012 Airfoil GAMM Case

The NACA 0012 airfoil was chosen for laminar calculations. The flow conditions are those of test case A2 of the GAMM Workshop on the Numerical Simulation of Compressible Navier-Stokes flows.[33] Laminar conditions are used as a simplified validation case for the hybrid grid coupling with non-trivial geometry. The free-stream conditions are: Mach=0.8, Temp=300 K, α=10.0 and Re_c=500 (where the chord length is one).

The test case is a steady-state problem and results were computed using LAVA and OVERFLOW for comparison. A CFL of 5.0 was used for all solvers with local time stepping. A structured O-grid was generated for the geometry consisting of 367 points on the airfoil, wall normal spacing of $2.0\mathrm{x}10^{-3}$ and an outer boundary 25 chord lengths away from the airfoil. For the hybrid approach of LAVA, a portion of the structured O-grid was utilized as the nearbody mesh and an off-body adaptive Cartesian background grid was utilized. The grids are illustrated in Figure 20. Viscous no-slip walls were specified on the airfoil and farfield conditions were specified on the domain boundaries. Both the Cartesian approach of LAVA and OVERFLOW were run using Roe flux-splitting with the minmod limiter. The unstructured approach was computed using the AUSPW+ flux-splitting and the minmod limiter.



(a) Structured Grid (b) Hybrid Grid

Figure 20. Computational grids for NACA 0012 GAMM test case using: (a) structured curvilinear and (b) hybrid grid approaches.

The surface pressure distribution on the airfoil is shown in Figure 21 with comparison to experimental data. There is excellent agreement between the two codes and experimental data. The coefficient of skin friction components were also computed and are plotted in Figure 22 using OVERFLOW and LAVA. Overall the results display the same trends and behaviors between the different solvers. One highlight of this test case was the use of delayed starting of the nearbody solver for the hybrid approach of LAVA. A steady state solution was obtained using the Cartesian approach of LAVA and an immersed representation of the NACA 0012 airfoil with adaptive mesh refinement before initiating the nearbody grid. The advantage of this approach is the ability to convect transient solutions on coarse Cartesian meshes and initialize the nearbody solution with a better solution. This procedure reduces the overall computational time of the simulation. The solution appeared highly sensitive to the off-body resolution of the shock on the upper surface of the airfoil. The final converged solution and adapted mesh is shown in Figure 23 for the hybrid approach.

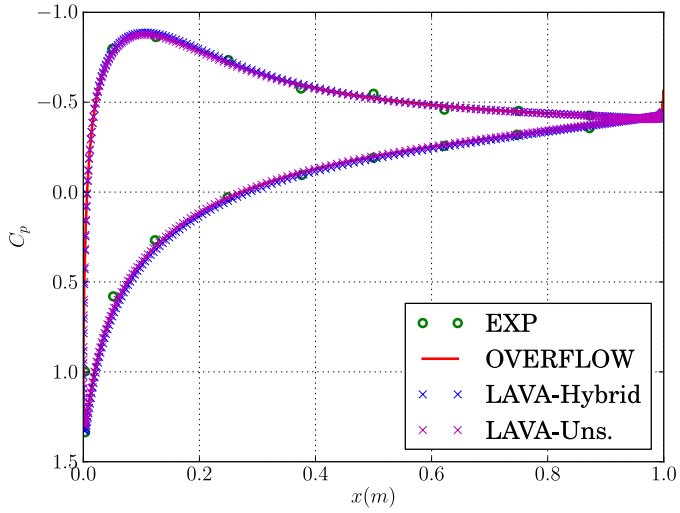
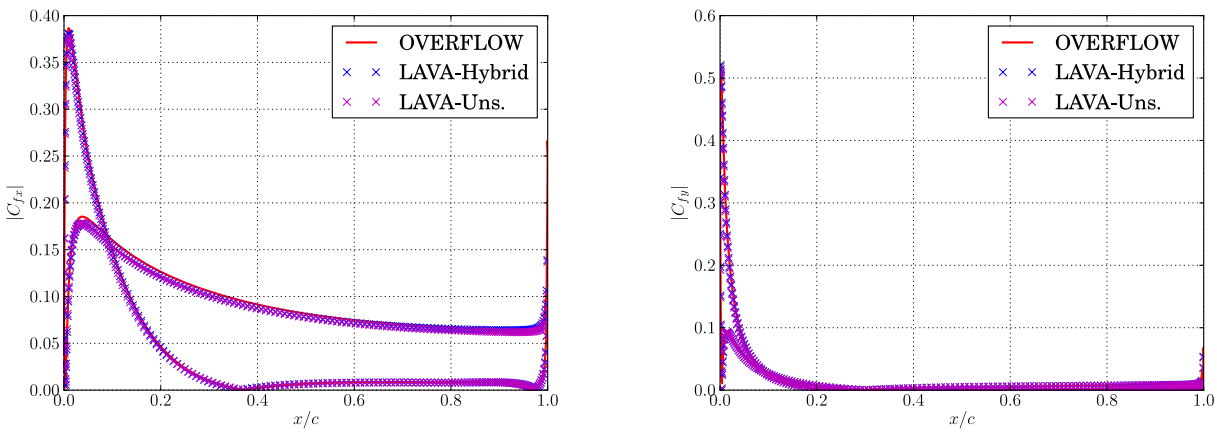


Figure 21. Pressure coefficient distribution on surface of NACA 0012 airfoil. Experimental data, hybrid and unstructured approaches of LAVA results are shown.



(a) $|C_{fx}|$ (b) $|C_{fy}|$

Figure 22. Coefficient of skin friction (a) x-component and (b) y-component distribution on surface of NACA 0012 airfoil using OVERFLOW, hybrid and unstructured approaches of LAVA.

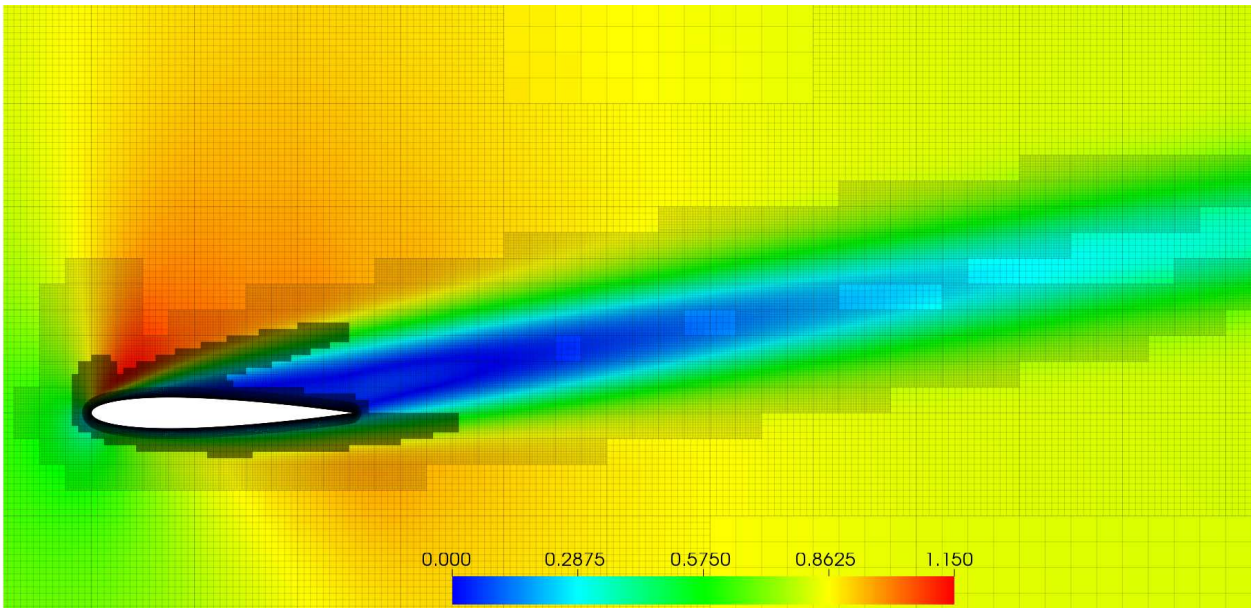


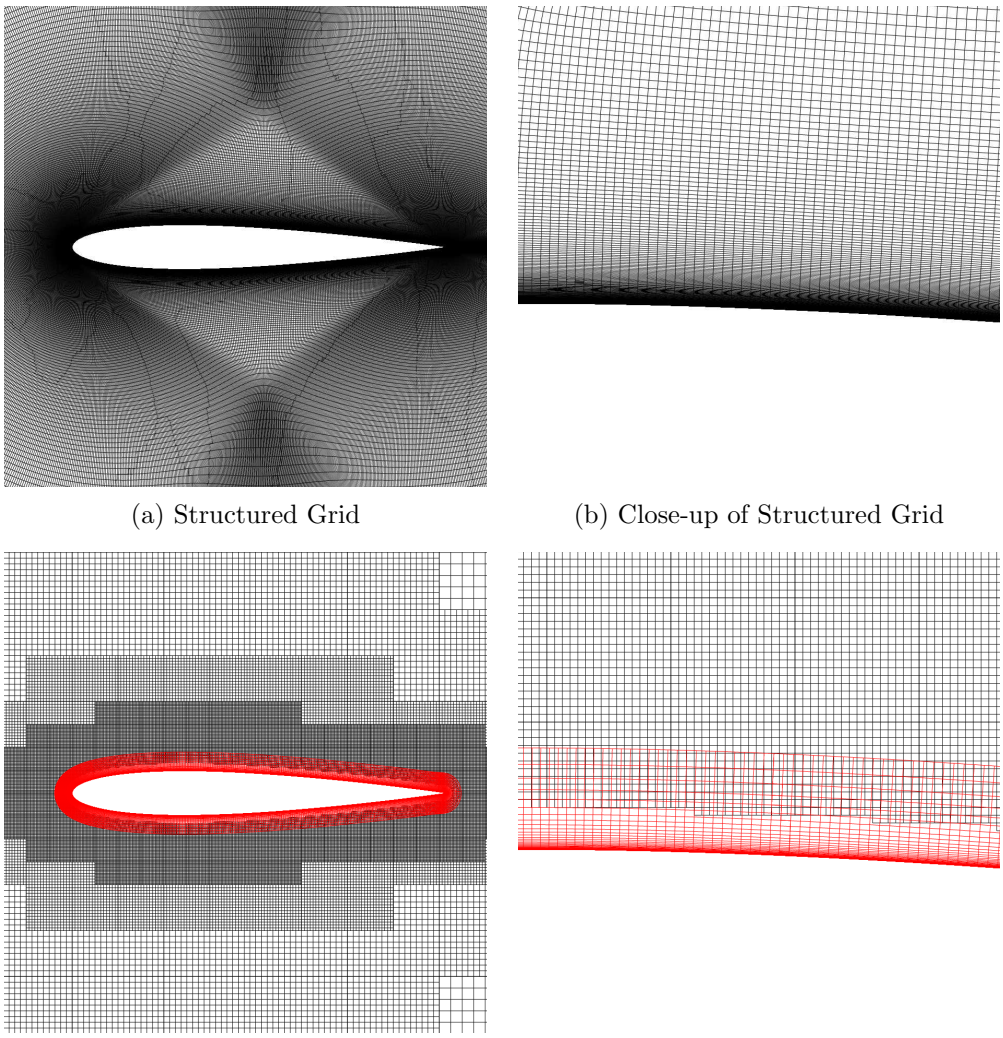
Figure 23. Mach number distribution for NACA GAMM case using hybrid approach with solution based adaption.

V.B. NACA 0012 Airfoil Case

The purpose of the NACA 0012 airfoil case is to validate LAVA and the turbulence models. This test case is also featured in the turbulence modeling resource[4,5] and has been applied to a variety of flow solvers. The test case uses essentially incompressible conditions (M=0.15 is used for compressible codes) and angles of attack from 0, 10 and 15 degrees. The free-stream conditions are: Mach=0.15, Temp=300 K, α=0 and Re_c= 6.0x10^6.

Both the LAVA and OVERFLOW solver were applied to this test case for validation and to assess the sensitivity to turbulence models. For all codes, the test case was run steady state with both the Spalart-Allmaras (SA-IA) and Mentor SST turbulence models (SST-2003). Central differencing was used in OVERFLOW, the Roe flux in the Cartesian approach of LAVA, and AUSMPW+ for the unstructured approach of LAVA.

A grid convergence study was completed to ensure an accurate solution was obtained. Five nested grid levels were analyzed using structured body fitted C-grids with the coarsest resolution having 113 points on the airfoil and 33 points in the wall normal direction. The finest resolution features 1793 points on the airfoil and 513 points in the wall normal direction. A hybrid overset grid was also developed using the same 4.6x10^{-7} wall normal spacing and airfoil spacing as the finest structured mesh. The finest level computational grids used for this test case are shown in Figure 24 for both the structured and hybrid approaches.



(a) Structured Grid (b) Close-up of Structured Grid

(a) Hybrid Grid (b) Close-up of Hybrid Grid

Figure 24. Structured grid for transonic NACA 0012 airfoil test case.

To assess the quality of the solutions, the force coefficients were analyzed on the surface of the airfoil. The computed coefficient of lift and drag are summarized in Table 2 for the SA turbulence model and Table 3 for the SST turbulence model. A small discrepancy between SA and SST solutions are observed for the various angles of attack. Good comparison is seen between the coefficient of lift and drag for the different solvers and grid strategies.

Method	C_L		C_D		
	$\alpha = 10$	$\alpha = 15$	$\alpha = 0$	$\alpha = 10$	$\alpha = 15$
OVERFLOW	1.08753	1.54332	0.00820	0.01236	0.02133
LAVA-Unstructured	1.08987	1.54762	0.00814	0.01235	0.02124
LAVA-Hybrid	1.11496	1.58327	0.00872	0.01444	0.02440

Table 2. Coefficient of lift and drags for various angles of attack using the SA turbulence model.

Method	C_L		C_D		
	$\alpha = 10$	$\alpha = 15$	$\alpha = 0$	$\alpha = 10$	$\alpha = 15$
OVERFLOW	1.08741	1.49001	0.00820	0.01259	0.02315
LAVA-Unstructured	1.07519	1.50104	0.00808	0.01249	0.02250
LAVA-Hybrid	1.10033	1.53718	0.00861	0.01415	0.02498

Table 3. Coefficient of lift and drags for various angles of attack using the SST turbulence model.

The force convergence for the test case using the unstructured approach is illustrated in Figure 25 for the range of angles of attack and grid resolutions. Excellent grid convergence is observed for the range of grid analyzed, and even the coarse level grids have good agreement in terms of the coefficient of lift. Drag predictions are more sensitive to the grid resolution but also converge towards a grid independent solution.

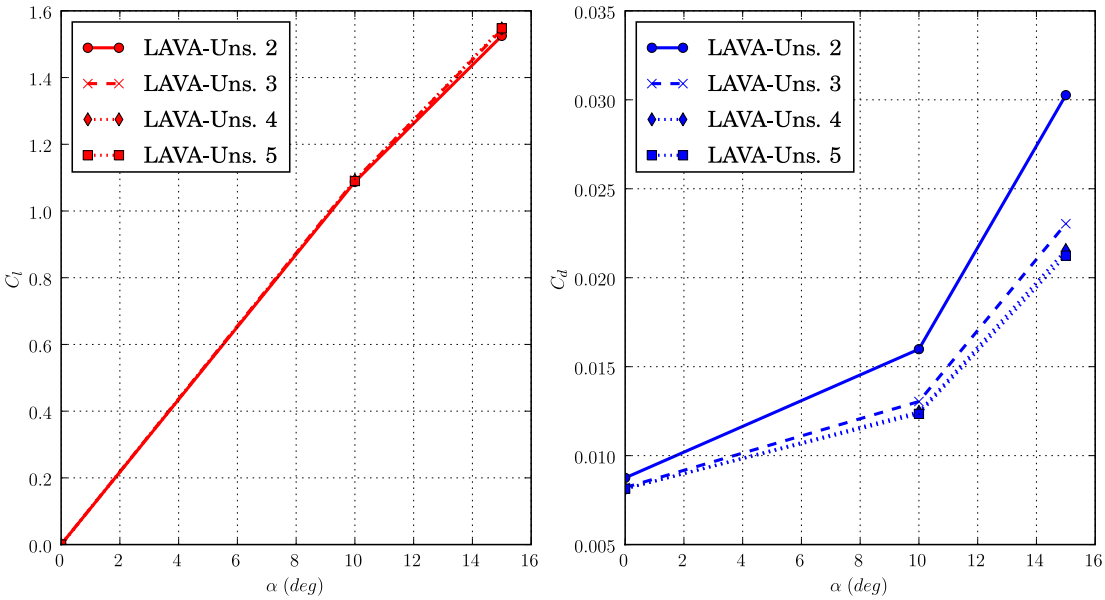
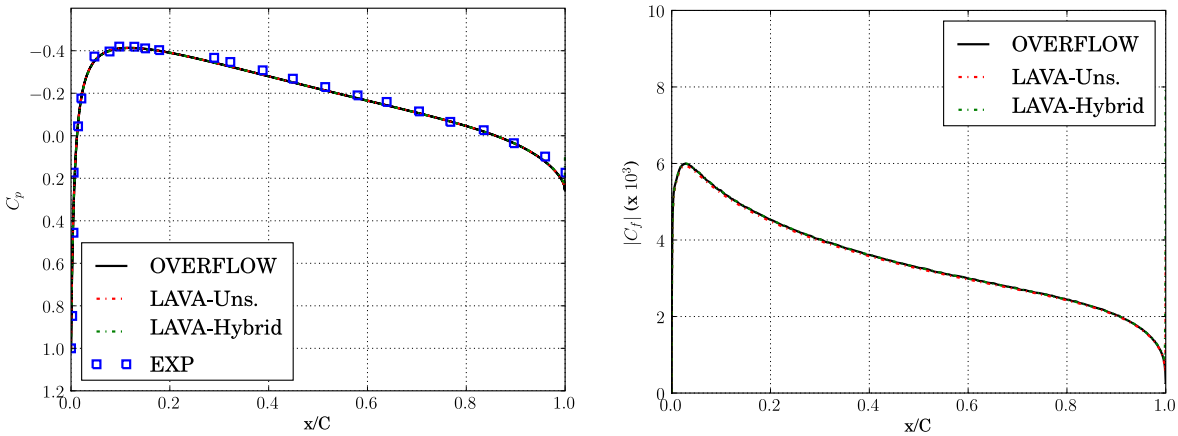


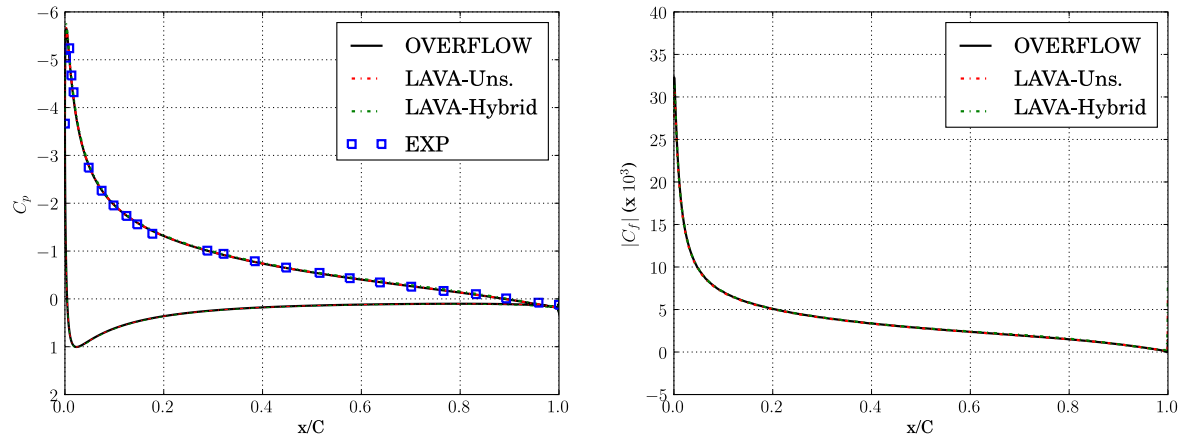
Figure 25. Unstructured approach of LAVA coefficient of lift and drag convergence for angle of attack range from 0-15 using multiple grid resolutions with the SA turbulence model for NACA0012 test case

The flow is characterized by a stagnation point at the leading edge and a large wake at the trailing edge. Especially for the high angle of attack case the turbulence models produce a large amount of eddy viscosity. Additional analysis was done by comparing the surface pressure distribution and skin friction on the airfoil with experimental data.[34] Figure 26 and Figure 27 show the pressure and coefficient of friction for the SA and SST turbulence models respectively. There is good agreement between the three approaches and experimental data. For all angles of attack the numerical results are in agreement as well. There is slightly

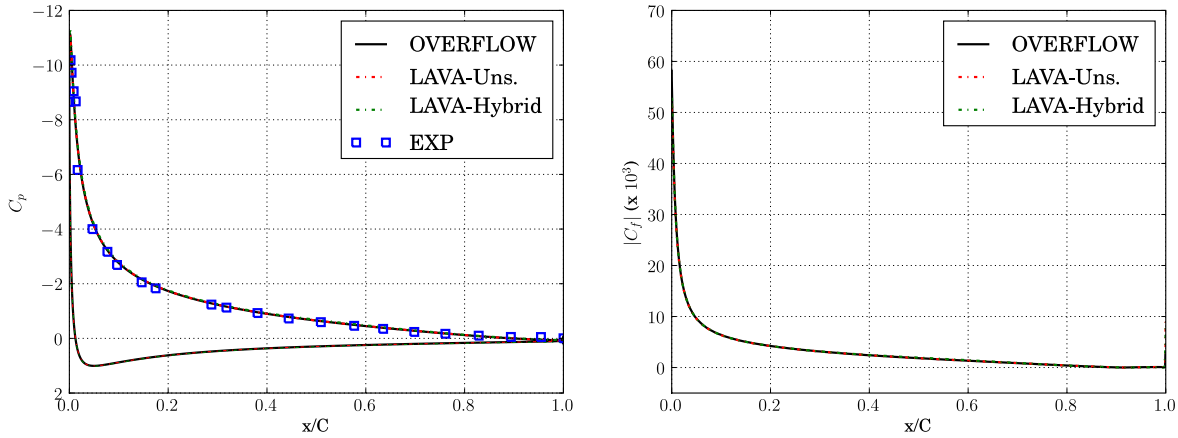
different behavior at the leading edge of the airfoil at zero angle of attack between turbulence models which becomes visible in the skin friction (see the slight dip at the leading edge skin friction for SST). One possible reason for this behavior is the difference in the turbulence model formulations near stagnation points.



(a) $\alpha = 0^o$

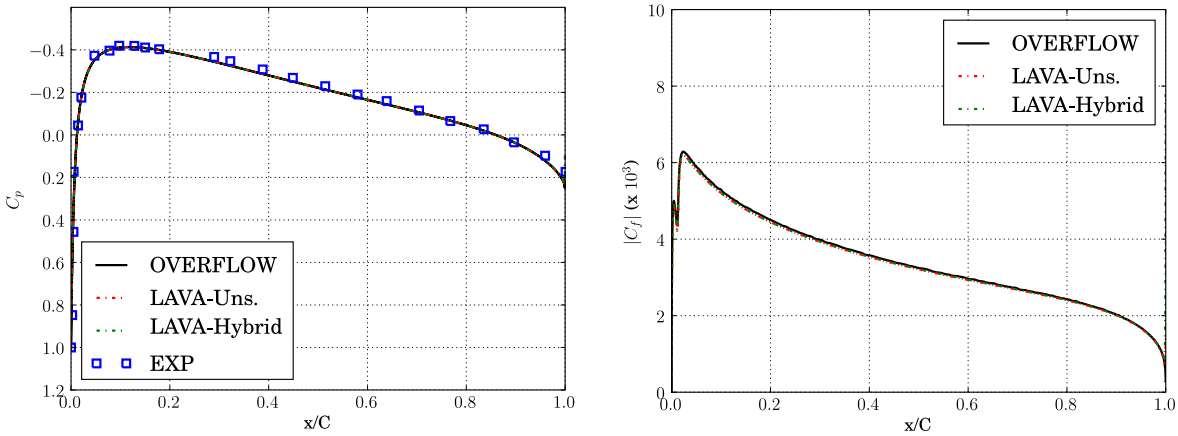


(b) $\alpha = 10^o$

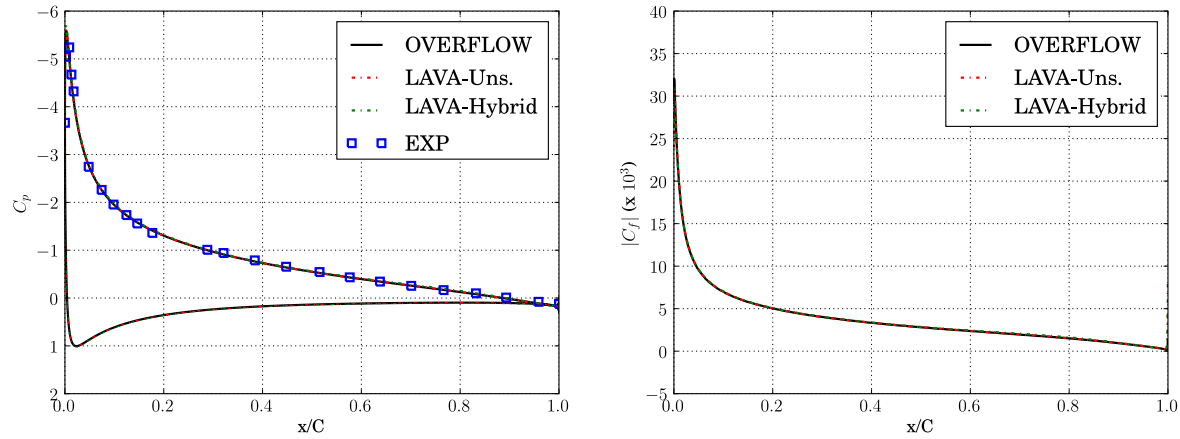


(c) $\alpha = 15^o$

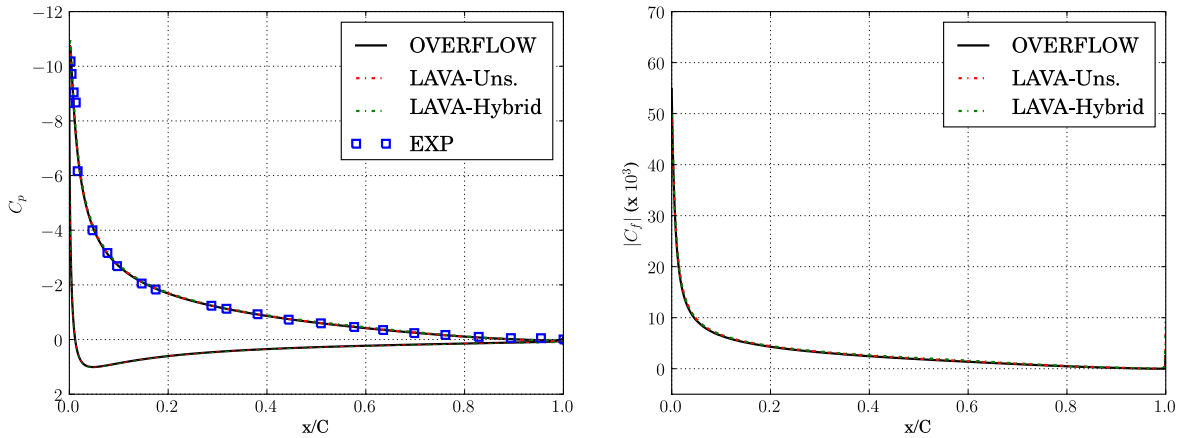
Figure 26. Coefficient of pressure and friction on the airfoil surface using SA turbulence model for NACA0012 test case at angle of attack: (a) 0, (b) 10 and (c) 15. Experimental data, OVERFLOW, unstructured and hybrid approaches of LAVA results are shown.



(a) $\alpha = 0^o$



(b) $\alpha = 10^o$



(c) $\alpha = 15^o$

Figure 27. Coefficient of pressure and friction on the airfoil surface using SST turbulence model for NACA0012 test case at angle of attack: (a) 0, (b) 10 and (c) 15. Experimental data, OVERFLOW, unstructured and hybrid approaches of LAVA results are shown.

V.C. ONERA M6

This test case corresponds to the transonic 3D ONERA M6 test case of the AGARD report[35] for which experimental results for coefficient of pressure are available. The test case is also part of the achieved test cases of NPARC Alliance Validation study.[36] The free-stream conditions are: Mach=0.8395, Temp=262.78, α=3.06 deg and Re_x=18.1x10^6. Both the unstructured approach of LAVA and OVERFLOW were solved

on this problem for validation and to assess the sensitivity to turbulence models. The codes were run with the Spalart-Allmaras (SA-IA) and Mentor SST turbulence models (SST-2003) on a fine structured C-H grid. The grid is shown in Figure 28 and features 203 points on the airfoil and 67 points in the normal direction. Central differencing was used with the Roe flux in OVERFLOW, while AUSMPW+ was used for unstructured approach.

The transonic flow conditions cause the typical "lambda" shock on the upper surface of the wing. The surface pressure distribution for unstructured approach is shown in in Figure 29 using the SA and SST models to illustrate the flow field. A comparison between the numerical and experimental[35] surface pressure distributions is shown in Figure 30 at multiple span locations. The lambda shock is visible in the surface pressure distribution slices and there is good agreement between the numerical and experimental data. A code to code comparison was also done for the skin friction coefficient components between unstructured approach and OVERFLOW. The x-component of the skin friction coefficient is shown at six span locations are shown in Figure 31 using both SA and SST turbulence models. The results indicate the SA and SST turbulence models are in good agreement with one another for OVERFLOW. There appears to be a larger discrepancy between the unstructured approach SA and SST results that requires further analysis. Overall the results display the same trends and behaviors between the two solvers.

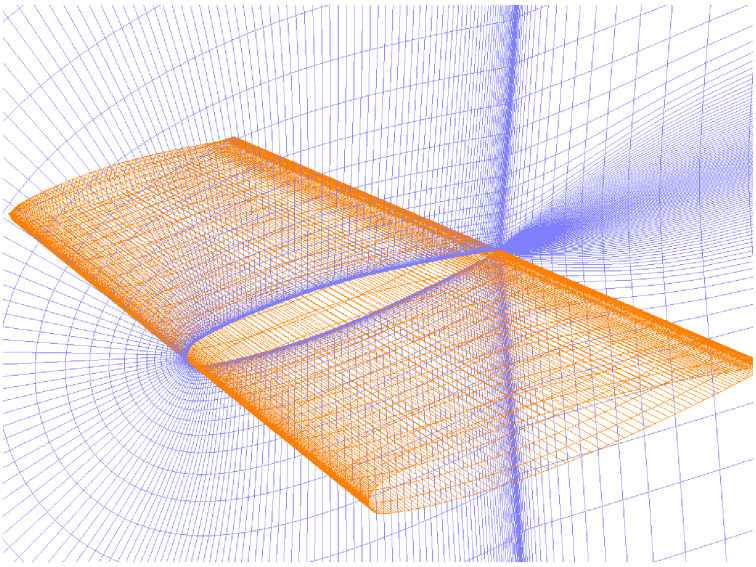
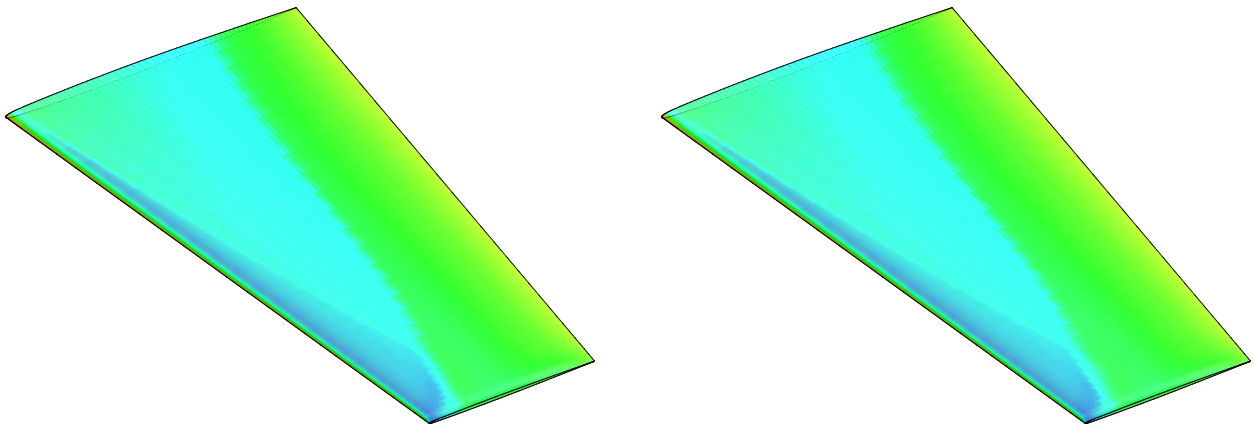
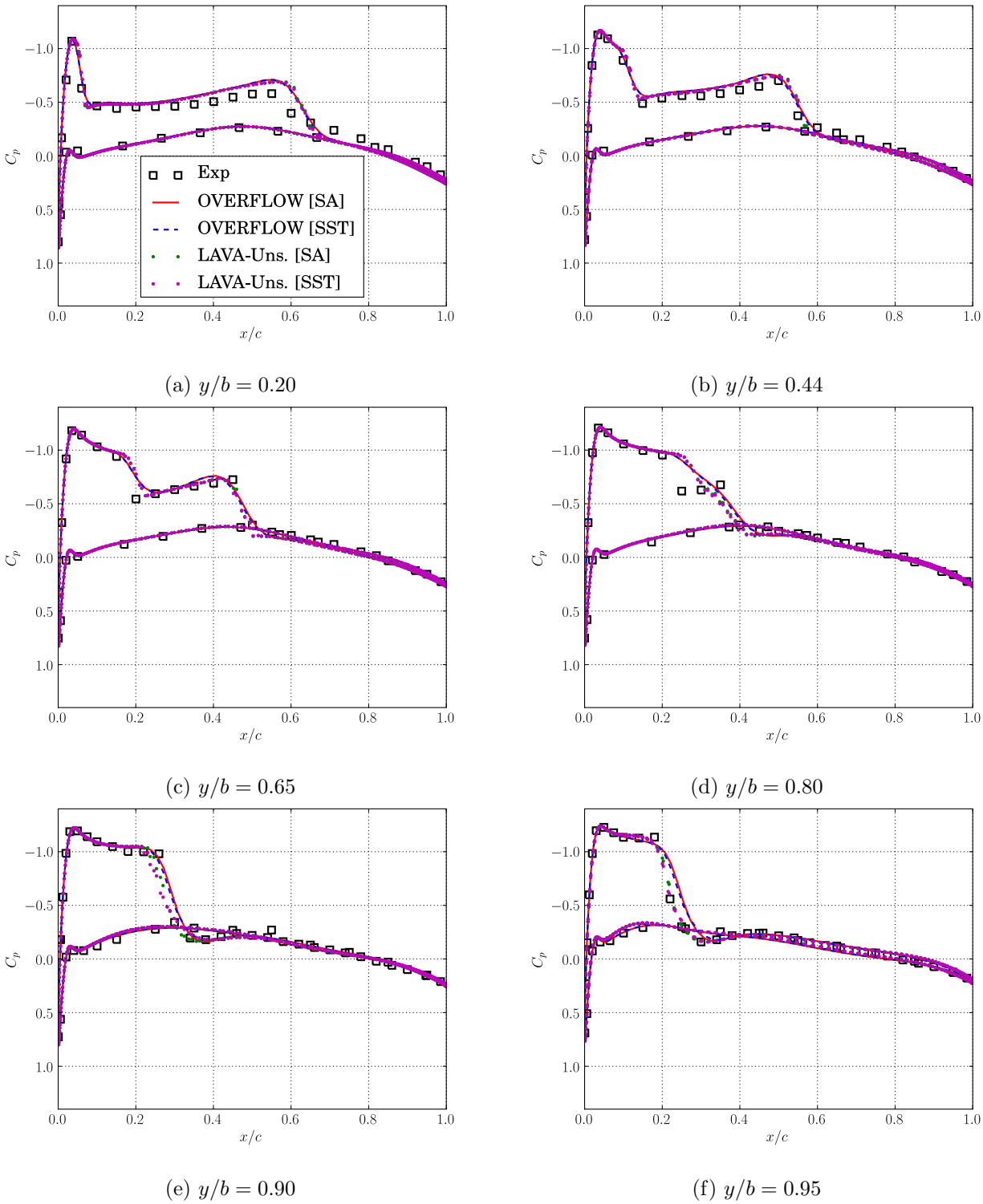


Figure 28. Structured C-H grid for transonic ONERA M6 airfoil test case.



(a) SA (b) SST

Figure 29. Surface pressure distribution for ONERA M6 test case using unstructured approach of LAVA with (a) SA and (b) SST turbulence models.

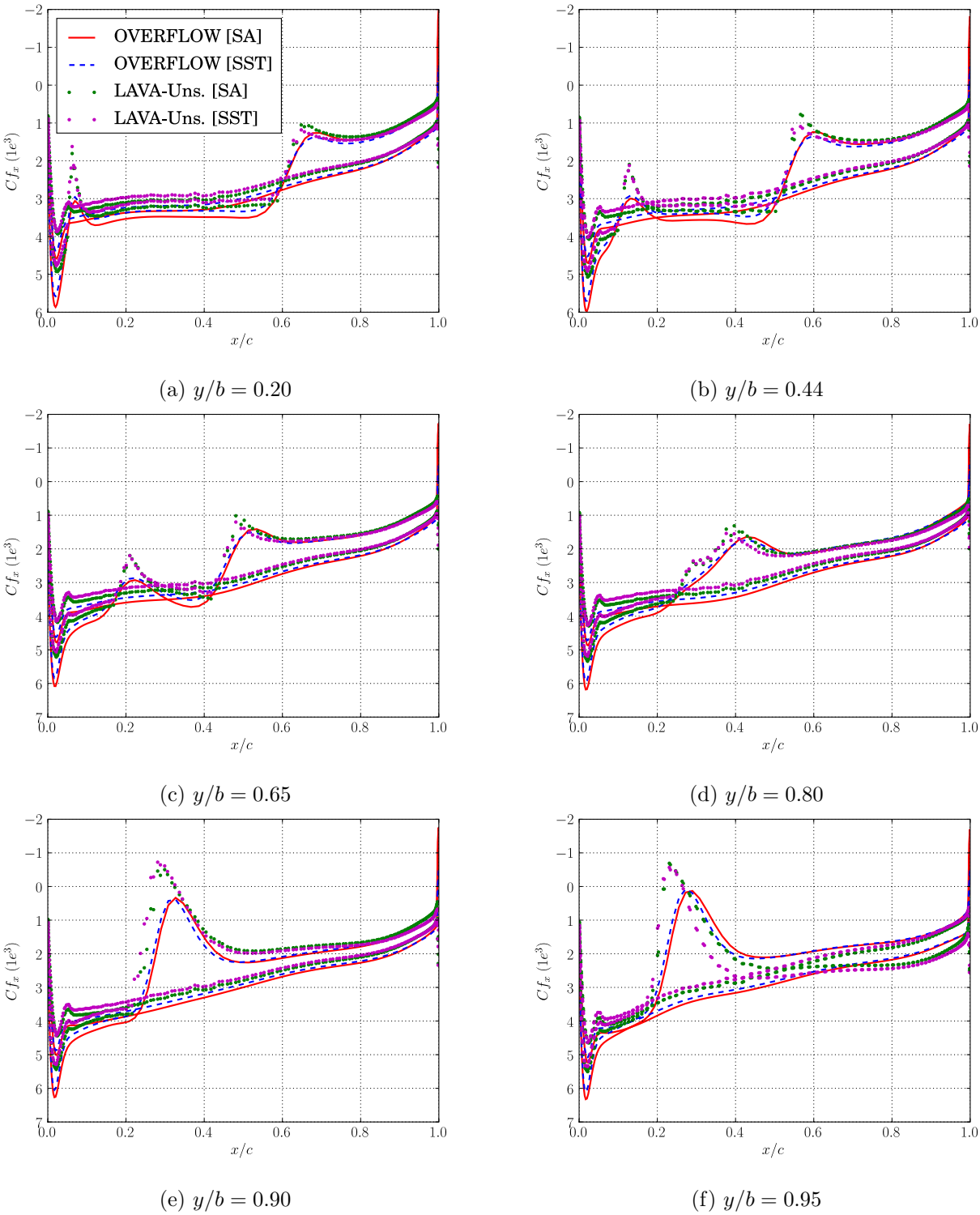


(a) $y/b = 0.20$ (b) $y/b = 0.44$

(c) $y/b = 0.65$ (d) $y/b = 0.80$

(e) $y/b = 0.90$ (f) $y/b = 0.95$

Figure 30. Pressure coefficient distribution on surface of ONERA M6 airfoil at multiple span locations ($b = 1.196$ m). unstructured approach SA (·), unstructured approach SST (·), OVERFLOW SA (–) and OVERFLOW SST (– –) results are shown.



(a) $y/b = 0.20$ (b) $y/b = 0.44$

(c) $y/b = 0.65$ (d) $y/b = 0.80$

(e) $y/b = 0.90$ (f) $y/b = 0.95$

Figure 31. X-component of skin friction coefficient distribution on surface of ONERA M6 airfoil at multiple span locations (b = 1.196 m). unstructured apprach SA (·), unstructured approach SST (·), OVERFLOW SA (–) and OVERFLOW SST (– –) results are shown.

VI. Conclusions

A pragmatic approach for the verification and validation of the LAVA CFD solver has been shown. Utilizing tools such as version control and continuous integration, the typically tedious tasks of preparation, verification, validation and documentation have been simplified and semi-automated. This procedure has been applied to LAVA with applications to Cartesian, unstructured and hybrid grid paradigms successfully. In this work, the results from MMS method for the verification of the 2D Euler and 3D Navier Stokes were shown. Systematic grid refinement was completed for both Cartesian and unstructured type grids. Overall the results indicate second-order spatial accuracy for both Cartesian and unstructured approaches of LAVA as expected.

Validation results from inviscid vortex propagation, NACA 0012 airfoil and ONERA wing also show good comparison with experimental and numerical data. With these sets of verification and validation test cases, the implementation and numerical accuracy of the individual Cartesian and unstructured approaches in addition to the hybrid approach of LAVA have been demonstrated. This automated procedure has proven effective at ensuring validity and accuracy of the solvers throughout the development process. Furthermore, the procedure allows for a collaborative and modular infrastructure to exist for productive and efficient code development. While it is nearly impossible to guarantee "bug-free" codes, the approach presented here represents a systematic approach in that direction. Future work will focus on expanding the verification and validation library to include more complex cases and refining the error metrics for individual test cases.

References

[1]American Institute of Aeronautics and Astronautics, "AIAA Guide for the Verification and Validation of Computational Fluid Dynamics Simulations," 1998, G–077–1998e.

[2]"Guide for Verification and Validation in Computational Solid Mechanics," *American Society of Mechanical Engineers, New York, NY*, ASME V & V 10-2006.

[3]Stern, F., Wilson, R. V., Coleman, H. W., and Paterson, E. G., "Verification and Validation of CFD Simulations," IIHR Report 407, Iowa Institute of Hydraulic Research, September 1999.

[4]Rumsey, C. L., "Consistency, Verification, and Validation of Turbulence Models for Reynolds-Averaged Navier-Stokes Applications," *3rd European Conference for Aerospace Sciences*, 2009.

[5]Rumsey, C. L., Smith, B. R., and Huang, G. H., "Description of a Website Resource for Turbulence Modeling Verification and Validation," *40th AIAA Fluid Dynamics Conference and Exhibit*, June 28–July 1 2010, AIAA–2010–4742.

[6]Roache, P., *Verification and Validation in Computational Science and Engineering*, Hermosa Publishers, 1998.

[7]Strawn, R. C., "Software Design Strategies for Multidisciplinary Computational Fluid Dynamics," *ICCFD7-3102, July 9-13, Big Island, Hawaii*, 2012.

[8]Housman, J., Barad, M., and Kiris, C., "Space-Time Accuracy Assessment of CFD Simulations for the Launch Environment," *29th AIAA Applied Aerodynamics Conference*, June 2011, AIAA-2011-3650.

[9]Moini-Yekta, S., Barad, M., Sozer, E., Brehm, C., Housman, J., and Kiris, C., "Towards Hybrid Grid Simulations of the Launch Environment," *ICCFD7-3102, July 9-13, Big Island, Hawaii*, 2012.

[10]"Apache Subversion," November 2012, http://subversion.apache.org/.

[11]"Jenkins CI," November 2012, http://jenkins-ci.org/.

[12]Housman, J., Kiris, C., and Hafez, M., "Time-Derivative Preconditioning Methods for Multicomponent Flows - Part I: Riemann Problems," *Journal of Applied Mechanics*, Vol. 76, No. 2, February 2009.

[13]Housman, J., Kiris, C., and Hafez, M., "Time-Derivative Preconditioning Methods for Multicomponent Flows - Part II: Two-Dimensional Applications," *Journal of Applied Mechanics*, Vol. 76, No. 3, March 2009.

[14]Spalart, S. and Allmaras, S., "A One-Equation Turbulence Model for Aerodynamic Flows," *30th Aerospace Sciences Meeting and Exhibit, Reno, NV*, January 1992, AIAA–92–0439.

[15]Menter, F., "Zonal Two Equation k-ω Turbulence Models For Aerodynamic Flows," *23rd Fluid Dynamics, Plasmadynamics, and Lasers Conference, Orlando, FL*, July 1993, AIAA–93–2906.

[16]Spalart, P., Jou, W., Strelets, M., and Allmaras, S., "Comments on the feasibility of LES for wings, and on a hybrid RANS/LES approach," *In Advances in DNS/LES, ed. C Liu, Z Liu, Columbus*, 1997, OH: Greyden Press.

[17]Spalart, P., Deck, S., Shur, M., Squires, K., Strelets, M., and Travin, A., "A new version of detached-eddy simulation, resistant to ambiguous grid densities," *Theo. and Comp. Fluid Dynamics*, Vol. 20(3), 2006, pp. 181–195.

[18]Kim, K. H., Kim, C., and Rho, O.-H., "Methods for the Accurate Computations of Hypersonic Flows: I. AUSMPW+ Scheme," *Journal of Computational Physics*, Vol. 174, 2001, pp. 38–80.

[19]Berger, M. J. and Colella, P., "Local Adaptive Mesh Refinement for Shock Hydrodynamics," *J. Comput. Phys.*, Vol. 82, No. 1, May 1989, pp. 64–84.

[20]Almgren, A. S., Bell, J. B., Colella, P., Howell, L. H., and Welcome, M. L., "A Conservative Adaptive Projection Method for the Variable Density Incompressible Navier-Stokes Equations," *J. Comp. Phys.*, Vol. 142, 1998, pp. 1–46.

[21]Barad, M. F. and Colella, P., "A Fourth-Order Accurate Local Refinement Method for Poisson's Equation," *J. Comp. Phys.*, Vol. 209, No. 1, October 2005, pp. 1–18.

[22]Barad, M. F., Colella, P., and Schladow, S. G., "An Adaptive Cut-Cell Method for Environmental Fluid Mechanics," *Int. J. Numer. Meth. Fluids*, Vol. 60, No. 5, 2009, pp. 473–514.

[23]Colella, P., Graves, D. T., Ligocki, T. J., Martin, D. F., Modiano, D., Serafini, D. B., and Straalen, B. V., "Chombo Software Package for AMR Applications - Design Document," unpublished.

[24]Mittal, R., Dong, H., Bozkurttas, M., Najjar, F., Vargas, A., and von Loebbecke, A., "A versatile sharp interface immersed boundary method for incompressible flows with complex boundaries," *Journal of Computational Physics*, Vol. 227, No. 10, 2008, pp. 4825 – 4852.

[25]Veluri, S. P., *Code Verification and Numerical Accuracy Assessment for Finite Volume CFD Codes*, Ph.D. thesis, Virginia Polytechnic Institute and State University, 2010.

[26]Oberkampf, W. L. and Roy, C. J., *Verification and Validation of Scientific Computing*, Cambridge University Press, Cabridge, 2010.

[27]Ahmad, F. and Al-Barakati, W. H., "An approximate analytic solution of the Blasius problem," *Communcations in Nonlinear Science and Numerical Simulation*, 2008, NASA TM-1314.

[28]Wieghardt, K. and Tillman, W., "On the Turbulent Friction Layer for Rising Pressure," 1951, NASA TM-1314.

[29]Coles, D. E., "Proceedings of the Computation of Turbulent Boundary Layers," *1968 AFOSR-IFP-Stanford Conference*, edited by D. E. Coles and E. A. Hirst, Vol. II, Thermosciences Division, Department of Mechanical Engineering, Stanford University Press, Stanford, Calif, 1969.

[30]Nichols, R. and Buning, P., "User's Manual for OVERFLOW 2.1," Version 2.1t.

[31]White, F., *Viscous Fluid Flow*, McGraw Hill, 1974.

[32]Pulliam, T., "High Order Accurate Finite-Difference Methods: as seen in OVERFLOW," *AIAA Journal*, June 2011, AIAA-2011-3851.

[33]"Numerical Simulation of Compressible Navier-Stokes Flows," *Notes on Numerical Fluid Mechanics*, edited by M. Bristeau, R. Glowinski, J. Periaux, and H. Viviand, Vol. 18, 1986, Proceedings of the GAMM workshop on the Numerical Simulation of Compressible Navier-Stokes flows, held at INRIA, Sophia-Antipolis (France), December 1985.

[34]Gregory, N. and O'Reilly, C. L., "Low-Speed Aerodynamic Characteristics of NACA 0012 Aerofoil Sections, including the Effects of Upper-Surface Roughness Simulation Hoar Frost," Jan 2008, NASA R&M 3726.

[35]Cook, P., McDonald, M., and Firmin, M., "Aerofoil RAE 2822 - Pressure Distributions, and Boundary Layer and Wake Measurements," *AGARD Report AR 138*, 1979.

[36]Slater, J. W., "NPARC Alliance Validation Archive," http://http://www.grc.nasa.gov/WWW/wind/valid/raetaf/raetaf.html/, November 2012, RAE2822 Transonic Airfoil Study 4.