LA-UR-12-21165
Approved for public release; distribution is unlimited.

Title: Code Verification of the HIGRAD Computational Fluid Dynamics Solver

Author(s): Van Buren, Kendra L.
Canfield, Jesse M.
Hemez, Francois M.
Sauer, Jeremy A.

Intended for: Report to LDRD Office
Report

Code Verification of the HIGRAD Computational Fluid Dynamics Solver

Kendra L. Van Buren [†]
Clemson University
Civil Engineering Department
Clemson, South Carolina

Jesse M. Canfield [&]
Los Alamos National Laboratory
EES-Division (EES-16)
Los Alamos, New Mexico

François M. Hemez [#]
Los Alamos National Laboratory
XTD-Division (XTD-3)
Los Alamos, New Mexico

Jeremy A. Sauer [$]
Los Alamos National Laboratory
EES-Division (EES-16)
Los Alamos, New Mexico

Executive Summary

The purpose of this report is to outline code and solution verification activities applied to HIGRAD, a Computational Fluid Dynamics (CFD) solver of the compressible Navier-Stokes equations developed at the Los Alamos National Laboratory, and used to simulate various phenomena such as the propagation of wildfires and atmospheric hydrodynamics. Code verification efforts, as described in this report, are an important first step to establish the credibility of numerical simulations. They provide evidence that the mathematical formulation is properly implemented without significant mistakes that would adversely impact the application of interest. Highly accurate analytical solutions are derived for four code verification test problems that exercise different aspects of the code. These test problems are referred to as: (i) the quiet start, (ii) the passive advection, (iii) the passive diffusion, and (iv) the piston-like problem. These problems are simulated using HIGRAD with different levels of mesh discretization and the numerical solutions are compared to their analytical counterparts. In addition, the rates of convergence are estimated to verify the numerical performance of the solver. The first three test problems produce numerical approximations as expected. The fourth test problem (piston-like) indicates the extent to which the code is able to simulate a "mild" discontinuity, which is a condition that would typically be better handled by a Lagrangian formulation. The current investigation concludes that the numerical implementation of the solver performs as expected. The quality of solutions is sufficient to provide credible simulations of fluid flows around wind turbines. The main caveat associated to these findings is the low coverage provided by these four problems, and somewhat limited verification activities. A more comprehensive evaluation of HIGRAD may be beneficial for future studies. *(Report approved for unlimited, public release on May xx, 2012, LA-UR-12-xxxx.)*

[†] Doctoral student, Clemson University. Mailing address: Department of Civil Engineering, Clemson University, Lowry Hall, Box 340911, Clemson, SC 29634-0911. E-mail: klvan@clemson.edu.

[&] Technical staff member, EES-Division. Mailing address: Los Alamos National Laboratory, EES-16, Mail Stop J535, Los Alamos, NM 87545. E-mail: jessec@lanl.gov.

[#] Technical staff member, XTD-Division. Mailing address: Los Alamos National Laboratory, XTD-3, Mail Stop T087, Los Alamos, NM 87545. E-mail: hemez@lanl.gov.

[$] Technical staff member, EES-Division. Mailing address: Los Alamos National Laboratory, EES-16, Mail Stop J535, Los Alamos, NM 87545. E-mail: jsauer@lanl.gov.

1. Introduction

This report details a preliminary effort to organize verification activities applied to HIGRAD, a Computational Fluid Dynamics (CFD) solver developed at the Los Alamos National Laboratory (LANL). This code is used to simulate various phenomena such as the propagation of wildfires and atmospheric hydrodynamics. HIGRAD is developed to take advantage of massively parallel computing architectures. The fluid dynamics solver is common to several applications, such as the simulation of wildfires (with the code FIRETEC), tracer transport and dispersion in urban areas, meteorological phenomena, and wind plants (with the code WindBlade) [1].

HIGRAD is used to simulate the performance of wind plants as part of the Laboratory Directed Research and Development Directed Research (LDRD-DR) project "Intelligent Wind Turbines" at LANL. The code WindBlade is being developed to couple the fluid dynamics solver HIGRAD with NLBeam, a nonlinear Finite Element (FE) representation of the structural components of wind turbines [2]. Using HIGRAD to develop this capability is advantageous because it can include atmospheric and topographic effects at the wind plant scale. The ultimate objective is to use WindBlade to economically study the power output of wind plants, optimal site placement, and array and wake impacts from turbine-turbine interactions and wind-turbine interactions.

The reliance on the HIGRAD solver in the development of WindBlade provides one of the first viable simulation efforts to study wind turbines at the plant scale. This unique capability captures the realistic topology of the terrain; implements boundary conditions that do not rely on cyclic symmetry; and is capable of performing highly resolved, three-dimensional simulations that are inaccessible to other wind turbine software. However, ***credible*** predictions for the performance of wind plants require a rigorous verification of the code used to develop these models.

Code verification is generally considered to be the first step of any Verification and Validation (V&V) study performed to establish the credibility of numerical simulations. Verification activities discussed in this report are different from the Software Quality Assurance (SQA) efforts typically carried out in a large-scale code project. SQA practices guarantee that the implementation of a model, solver or algorithm meets requirements of quality and performance from the perspective of the code developer. An example in the field of FE modeling is the implementation of "patch tests" to verify that a specific finite element possesses the correct number of rigid body modes and recovers the appropriate uniform-strain state. These practices are aimed at demonstrating the absence of programming mistakes. They do not, however, assess the performance of an algorithm, or numerical quality of discrete solutions, for a ***particular*** application of interest.

The verification efforts presented in this report are performed from the perspective of the code user. They are applied to a particular application of the HIGRAD solver, which is the numerical simulation of flow dynamics around wind turbines. The difference with SQA practices is that the test problems analyzed here are not attempting to ***unambiguously*** prove that the software is correctly implemented. Rather, the analysis is meant to demonstrate that the implementation is properly formulated without any significant "bugs" or programming mistakes that may adversely impact the application of interest. Herein, we are not attempting to "prove" correctness; rather, we are attempting to dispute the existence of a major mistake.

The importance of verification activities is emphasized from previous work on the development of FE models to simulate the structural vibration response of wind turbine blades. A recent code verification effort exposed a deficiency in the implementation of shell elements into commercial software [3]. Numerical estimates obtained from the FE analysis were compared to closed-form analytical solutions for a hollow cylinder subjected to torsional loading. The study found that the software incorrectly approximated the exact solution and, of even greater concern, that refining the mesh increased the error in the solution. Although simple, these findings called into question

the results of previous simulations in which these same shell elements were used to approximate the behavior of wind turbine blades. This example demonstrates the necessity of using code verification to establish credibility and verify the correctness of numerical models.

Because the verification discussed in this report is application-driven, the investigation starts by identifying phenomena that are most important in the simulation of flow dynamics around wind turbines. The main features of the HIGRAD solver are overviewed in Section 2. A short list of the phenomenology verified in this report is given in Section 3. This list is surely not exhaustive but it illustrates a systematic procedure that can be implemented to prioritize the verification effort, while avoiding spending time and precious resources on aspects of the simulation that may not matter for the application of interest.

Section 3 also discusses the verification activities performed that belong to two categories. The first category is the comparison of discrete solutions, obtained by simulating a test problem, to exact solutions. Metrics, such as the absolute difference between scalar quantities or Euclidean distances estimated over the computational domain, are calculated to assess the error between discrete and analytical solutions. This first category of verification, generally referred to as code verification, leads to a straightforward interpretation. The code either passes or fails the test. But it also presumes the availability of analytical solutions for comparison. Analytical solutions are available only for simple test problems that may not represent the complexity of scenarios that the software attempts to simulate.

The second category of activities consists of refinement studies to assess the extent to which the code self-converges as the level of resolution is increased. Refinement can be performed in either space or time. In addition to studying self-convergence, solutions of a refinement study are also used to assess the numerical performance of the hydrodynamics code. This technology is briefly overviewed in Section 3. This second category is referred to as solution verification and it offers the advantage that analytical solutions are not required to study self-convergence.

Sections 4, 5, 6, and 7 present the results of this preliminary verification study. The four test problems considered are the: (i) quiet start, (ii) passive advection, (iii) passive diffusion, and (iv) piston-like problems. All of these problems assess the extent to which the hydrodynamic solver accurately solves the conservation laws. The last problem stretches the domain of applicability of HIGRAD because it simulates the Rankine-Hugoniot condition of a discontinuity, a condition that is not expected to be encountered in the typical operation of a wind turbine. This problem is nevertheless studied to observe the extent to which stable solutions are calculated for a type of fluid flow outside the nominal operating regime of the software. High-level conclusions are given in Section 8 and details of the analytical solutions derived are presented in the appendix.

2. Overview of the HIGRAD Fluid Dynamics Solver

HIGRAD is a Computational Fluid Dynamics (CFD) model designed to represent sharp gradients in fluid flows. The acronym stands for "HIgh GRADient." The model explicitly solves the compressible Navier-Stokes equations in arbitrarily complex, three-dimensional Cartesian geometry [4]. The fundamental conservation law solved, written in Eulerian frame-of-reference, that is, which remains fixed in space and time, is:

$$\frac{\partial(\rho \mathrm{u})}{\partial t} + \mathrm{u} \cdot \nabla(\rho \mathrm{u}) = -\nabla \mathrm{p} + \nabla \cdot \sigma_{\mathrm{D}} + \mathrm{f}, \tag{1}$$

where the triplet (ρ; p; u) of state variables denotes the density, pressure, and velocity vector of an element of fluid, respectively. Symbols σ_D and f in the right-hand side denote the deviatoric stress tensor and applied body forces. The complete stress tensor, that includes isotropic and deviatoric components, is defined as:

$$\sigma = -\mathrm{p}\ \mathrm{I} + \sigma_{\mathrm{D}}. \tag{2}$$

The Navier-Stokes equation (1) expresses the conservation of momentum of a fluid element. It is augmented by the continuity equation that represents the conversation of mass during motion of the fluid:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathrm{u}) = 0, \tag{3}$$

and by the conservation of total energy. In one-dimensional "slab" geometry, and assuming that the fluid is inviscid and adiabatic, this system of equations reduces to the well-known Euler equations discussed in the appendix, and represented as:

$$\begin{cases} \dfrac{\partial \rho}{\partial \mathrm{t}} + \dfrac{\partial(\rho\ \mathrm{u})}{\partial \mathrm{x}} = 0 \\ \dfrac{\partial(\rho\ \mathrm{u})}{\partial \mathrm{t}} + \dfrac{\partial(\rho\ \mathrm{u}^2 + \mathrm{p})}{\partial \mathrm{x}} = 0 \\ \dfrac{\partial(\rho\ \mathrm{E})}{\partial \mathrm{t}} + \dfrac{\partial\ \mathrm{u}\ (\rho\ \mathrm{E} + \mathrm{p})}{\partial \mathrm{x}} = 0 \end{cases}, \tag{4}$$

where E denotes the total specific energy of an element of fluid, that is, $E = \varepsilon + \frac{1}{2} u^2$ (and ε is the internal specific energy). Equations (1) to (4) do not include the equation of state added to provide closure by relating density, pressure, and energy. In this work, and because the focus is on verifying the numerical simulation of air flowing around wind turbines, a perfect gas law with adiabatic exponent γ = 7/5 is used.

The HIGRAD code solves this system of coupled, nonlinear equations to calculate the values of density (ρ), pressure (p), velocity (u_X, u_Y, u_Z), and internal energy (ε) or temperature (T) in every zone of the computational grid. Depending on the problem, it offers the additional capability to solve for secondary variables such as the concentration of chemicals in the flow. This is used, for example, to simulate the dispersal of contaminants. The solver implements the finite volume method on a generalized coordinate grid. HIGRAD has been used to model hurricanes [5], cloud physics [6], and wildfires [7] among a number of diverse atmospheric phenomena.

The HIGRAD code discretizes the computational domain using hexahedron “cells” (or zones). Zones of the discretization are treated using the finite volume method, whereby state variables are approximated using a sub-grid model denoted as a generic function F(•) within each control volume $V^{(E)}$ of the mesh. For example, the density is approximated as:

$$\rho^{(E)} = \int_{V^{(E)}} F(\rho)\, d\rho\,, \tag{5}$$

where the integration of the sub-grid model is carried out over the volume $V^{(E)}$ of each zone. The HIGRAD code implements a piecewise constant sub-grid model, which means that F(•) = 1 and the state variables are constant within each finite volume of the discretization. With this choice, for example, equation (5) gives $\rho^{(E)}$ = constant in zone $V^{(E)}$.

A number of standard finite volume schemes are employed to represent the various terms in the governing equations. See, for example, those given in references [8] and [9]. The HIGRAD code includes several temporal integration and spatial differencing schemes. The advective terms are computed using a Flux Corrected Transport (FCT) algorithm, where QUICK or QUICKEST are used for the high-order scheme and upstream differencing is used for the low-order scheme [10]. The FCT algorithm limits the high-order flux calculations using low-order fluxes to ensure the positive-definite advection of prognostic variables. Simply speaking, it means that the flux of quantity, such as mass or momentum, flowing through a zone from its neighbors cannot exceed the cumulative amount present in the originating zones. The pressure gradient, diffusion, and turbulence terms are approximated by a second-order accurate, centered-in-space scheme.

The discretized equations are integrated in time explicitly. The schemes available for integration are Euler forward; Runge-Kutta of order-1, order-2, order-3, or order-4; Adams-Bashforth; and the Method of Averages (MOA) [11]. The Runge-Kutta family of time integration techniques is used exclusively for this study.

3. Rationale of the Selection of Verification Test Problems

To emphasize the importance of verification activities, the principle of discretization is briefly explained in this section. The key point is that, when a system of partial differential equations is solved numerically by a computer code, the discrete solutions obtained approximate a system of equations that is ***different*** from the original system of equations! Understanding this difference is essential to assess if the software is performing correctly and, therefore, if it can be trusted.

3.1 The Consistency and Convergence of Modified Equations

The HIGRAD fluid dynamics code solves conservation laws that balance the rate-of-change in time of the state variable, the gradient of its flux, and a potentially non-zero source term in the right-hand side. Without loss of generality, this equation can be written in a one-dimensional, Cartesian coordinate system as:

$$\frac{\partial y^{Exact}(x;t)}{\partial t} + \frac{\partial F(y^{Exact}(x;t))}{\partial x} = S(x;t), \tag{6}$$

where $y^{Exact}(x;t)$ is the exact solution, $F(\bullet)$ denotes the flux term, and $S(x;t)$ is a source or forcing function that drives the dynamics of the system. These generic functions depend on space and time, labeled "x" and "t," respectively. When solving equation (6) with a numerical method, one seeks the best-possible approximation of the exact solution y^{Exact}.

The numerical method discretizes the continuous equation (6) on a computational mesh to yield a discrete solution $y_k^n = y(k\cdot\Delta x; n\cdot\Delta t)$ where Δx and Δt are the spatial and temporal resolutions, respectively. The approximation y_k^n is obtained by solving a discretized equation that looks, for example, something like:

$$\frac{y_k^{n+1} - y_k^n}{\Delta t} + \frac{F_{k+1/2}^n - F_{k-1/2}^n}{\Delta x} = S_k^n. \tag{7}$$

The approximation of equation (7) corresponds to a hypothetical discretization scheme where the time differentiation operator ($\partial\bullet/\partial t$) is approximated by a forward Euler difference while the spatial differentiation operator ($\partial\bullet/\partial x$) is approximated by a centered difference. It is emphasized that equation (7) is shown for illustration only; it is not the differentiation scheme implemented in the HIGRAD code. The discrete values y_k^n, F_k^n, and S_k^n can be obtained from finite difference, finite volume, or finite element approximations. One observes that the discretized equation (7) "appears" similar to the original, continuous equation (6). The similarity, however, is misleading.

Contrary to common belief, the discrete solutions y_k^n are ***not*** approximations of the continuous solution y^{Exact} of equation (6). Using the technique known as Modified Equation Analysis (MEA), see References [12] and [13], it can be shown that the approximations y_k^n converge to the solution of a ***modified equation*** that takes a form such as:

$$\frac{\partial y^{MEA}(x;t)}{\partial t} + \frac{\partial F(y^{MEA}(x;t))}{\partial x} = S(x;t) + \left(\frac{\partial^2 y^{MEA}(x;t)}{\partial t^2}\cdot\Delta t + \frac{\partial^3 y^{MEA}(x;t)}{\partial x^3}\cdot\Delta x^2 + \ldots \right). \tag{8}$$

Note that this example is conceptual and the correct form of the modified equation depends on properties of the original equation (6) and numerical method (7) implemented for its resolution. What is important to the discussion is that the continuous solution y^{MEA} of the modified equation (8) is ***different*** from y^{Exact}, at least, as long as the spatial and temporal resolutions remain finite, that is, as long as $\Delta x \neq 0$ and $\Delta t \neq 0$. The Lax equivalence theorem of Reference [14] details the

conditions under which the discrete approximations y_k^n are consistent with, and converge to, the continuous solution y^{Exact}. These concepts are illustrated graphically in Figure 1.

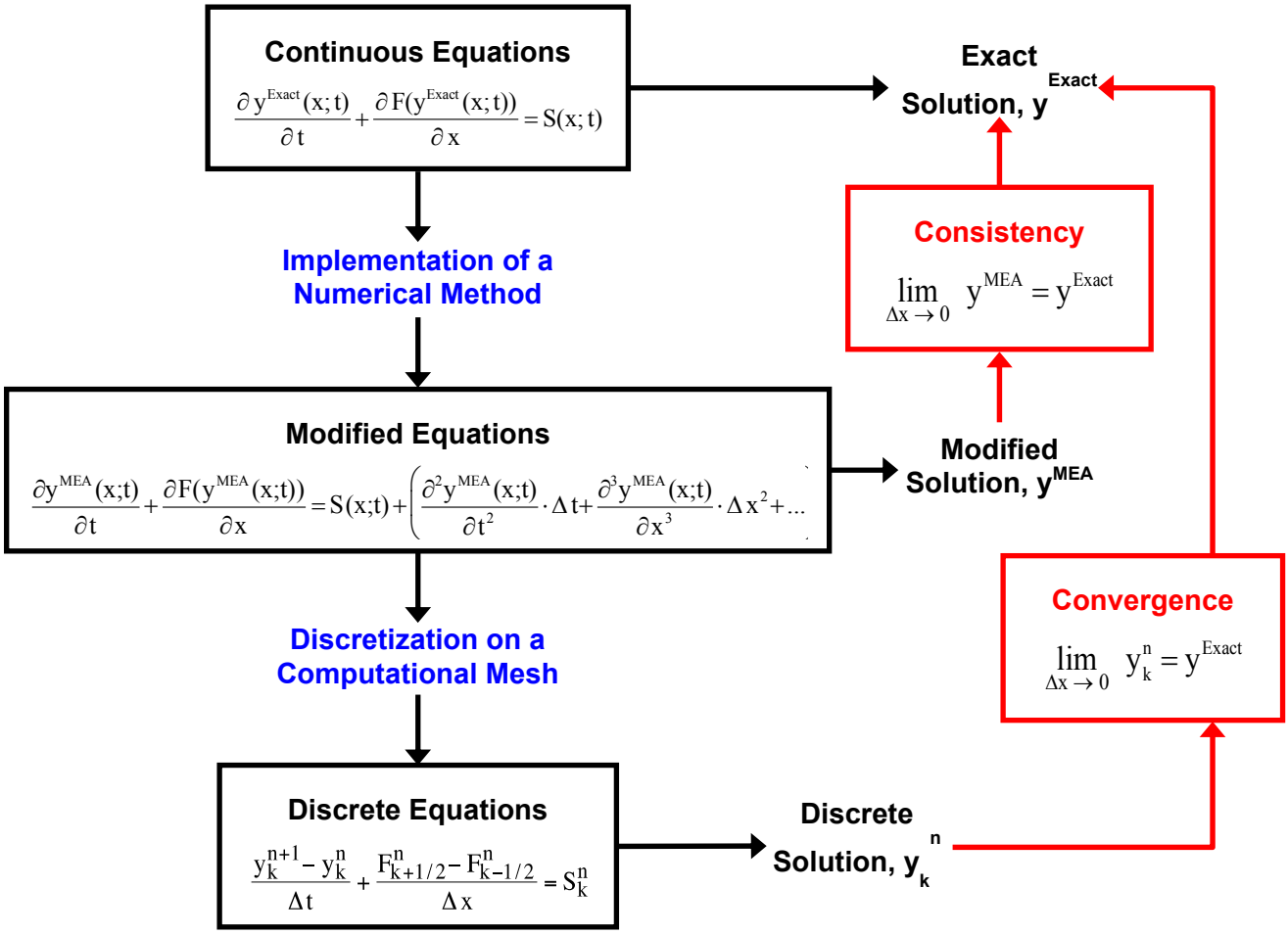

Figure 1. Concepts of consistency and convergence of numerical solutions.

The terms shown between parentheses in the modified equation (8) represent an infinite series expansion that characterizes the truncation error of the numerical simulation. It can be seen that truncation is what explains the difference between the continuous solutions y and y^{Exact}. Since the leading-order term of spatial truncation is proportional to an exponent, such as Δx or Δx^2, performing a mesh refinement study indicates how this error behaves. Likewise, performing a refinement study in time assesses how the temporal discretization error behaves.

3.2 The Regime of Asymptotic Convergence of Discrete Solutions

Because truncation error is the main mechanism by which the discrete solution y_k^n obtained by running the analysis software differs from the exact solution y^{Exact}, understanding its behavior is key to assessing the numerical performance of the code. Hence, verifying the quality of discrete solutions hinges on the concept of the ***regime of asymptotic convergence***.

Different choices of resolution Δx in a calculation induce different behaviors of the overall numerical error, as illustrated in Figure 2. By definition, the regime of asymptotic convergence is

the region where truncation effects dominate the overall production of numerical error. Plotting the solution error $||y^{Exact} - y_k^n||$ versus mesh size Δx gives a conceptual illustration of the main three regimes of the discretization.

Going from right (larger values of Δx) to left (smaller values of Δx), the first domain shown in color red is where the choice of mesh size is not appropriate to solve the discrete equations. This is, for example, the case when zones/cells are too coarse to resolve important geometrical features of the problem; when a constraint of numerical stability is violated; or when the mesh is too coarse to capture a characteristic scale over which the dynamics occur. Although it could be argued that discrete solutions originating from this regime of discretization are useful to indicate overall trends, their numerical quality should be questioned with the highest degree of suspicion.

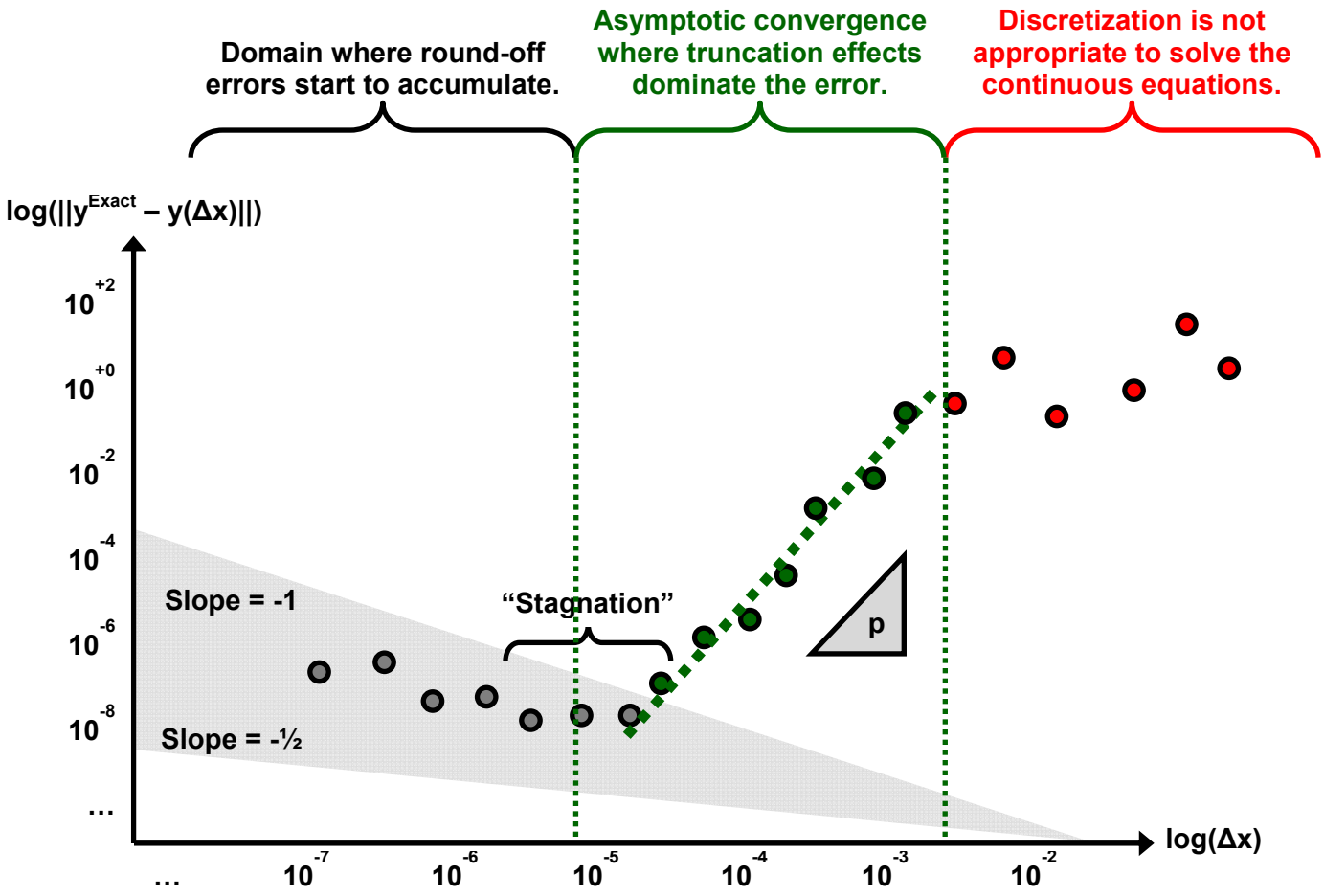

Figure 2. The three regimes of solution error versus spatial discretization.

The second region shown in color green in Figure 2 is where truncation dominates the overall production of numerical error; it is the regime of asymptotic convergence. In this second regime, the numerical error can be reduced, that is, solution accuracy can be improved, by increasing the level of resolution in the calculation. We have just outlined the basic principle of conducting a mesh refinement study.

The conceptual illustration shown in Figure 2 assumes that truncation effects, within the regime of asymptotic convergence, are dominated by a single error proportional to Δx^p, which appears as a straight line of slope "p" on a log-log representation. The particular functional form of the

modified equation, suggested in equation (8), motivates this assumption. It is also the reason why the behavior of truncation error is usually modeled by a simple equation such as:

$$\varepsilon(\Delta x) = \left\| y^{Exact} - y_k^n \right\| = \beta \, \Delta x^p + O\left(\Delta x^{p+1}\right), \tag{9}$$

where ε(Δx) denotes the difference, estimated in the sense of a user-defined norm ||•||, between the exact solution y^{Exact} of the continuous partial differential equations and the discrete solution y_k^n obtained at mesh size Δx. The pre-factor β represents a (constant) regression coefficient. The exponent p characterizes the rate at which the solution error ε(Δx) decreases as the level of resolution in the calculation is increased, or $\Delta x \rightarrow 0$. The value of exponent p should match the order of accuracy of the numerical method implemented in the code. For example, the second-order accurate Gudonov scheme of a finite volume method should exhibit a rate of p = 2.

The last region shown in color grey in Figure 2 is a limiting case for asymptotic convergence due to the fact that, with finite arithmetic implemented in a computer, round-off effects eventually start to accumulate as $\Delta x \rightarrow 0$. Round-off could then accumulate to the point where it supplants truncation error as the dominant mechanism that produces numerical error.

Understanding the regime of asymptotic convergence is important for three reasons. First, the analyses of mesh refinement performed in this work are valid only within the asymptotic regime. Second, verifying that the level of resolution Δx used in a calculation leads to discrete solutions within the asymptotic regime provides a strategy to reduce the solution error. If the error is too large for the intended application, then repeating the calculation with more resolution reduces it. This statement, however, is true only within the regime of asymptotic convergence. The third reason why it is important to understand the boundaries of the asymptotic regime, is that it is preferable to perform a calculation with the coarsest mesh that one can get away with. Coarser discretizations imply a lesser need for computational resources and faster turn-around times.

3.3 Rationale for the Selection of Code Verification Test Problems

The main objective of the verification work is to demonstrate that the numerical approximations provided by the HIGRAD code can be trusted for the simulation of wind turbine performance. To do so, the approximations are compared to exact solutions, or highly accurate solutions, derived from closed-form formulations of the conservation laws. These closed-form solutions, however, are available only for a limited number of simplified verification test problems and it is important to choose them without losing focus of the application of interest.

When exact solutions are available for comparison, it is straightforward to evaluate a metric of error. This is discussed in section 3.4. Code credibility is then established by demonstrating that the solution error remains low for an as-wide-as-possible range of test problems.

A second objective is to assess code performance. After verifying the lack of programming mistake that would significantly deteriorate the quality of numerical solutions, the question of code performance can be asked. Performance refers, here, to the overall behavior of truncation effects introduced by the implementation of specific algorithms or solvers. It is assessed through mesh refinement studies whereby the rate of convergence estimated from empirical solutions is compared to the formal order of accuracy of the numerical method. A code performs according to expectation if the accuracy that it provides compares favorably to the theory. This analysis is discussed in section 3.5 for the case where the exact solution of a test problem is known. For completeness, section 3.6 summarizes the case where the exact solution is not available.

Because resources devoted to this study are limited, it is not possible to analyze a large number of test problems. The problems analyzed, therefore, are carefully selected. For the application of

interest, it is deemed important to analyze test problems that assess the ability of the code to conserve mass, momentum, and energy while solving for state variables that represent the fluid flow around a wind turbine. A premium is placed on selecting test problems for their ability to diagnose a potential violation of the conservation laws.

A second consideration is the verification of the flux corrected transport, an algorithm whose implementation can significantly influence the numerical quality of discrete solutions. A third and final consideration originates from the application of the HIGRAD code to flow dynamics around wind turbines. Realistic wind conditions include, at a minimum, velocity gradients. It is desirable to devote one test problem to the ability of the code to represent these velocity gradients.

These considerations lead to the four test problems listed in Table 1. They comprise the rows of the table, while the columns identify which aspects (conservation law, flux corrected transport, and velocity gradient) are more specifically addressed by each problem. This way of selecting test problems is a simplified implementation of a Phenomenon Identification and Ranking Table (PIRT). A full PIRT analysis would consist of three basic steps that, here, have been simplified. The first step provides an as-exhaustive-as-possible list of the phenomenology that must be modeled for the application of interest. The phenomena include conservation laws, algorithms, models, and any other modeling or numerical aspect needed for the simulation. The second step ranks these phenomena from the most to least important. The goal is to prioritize the effort and avoid spending valuable resources to verify aspects to which predictions of the code may not be sensitive. The third and final step is to develop test problems appropriate to assess the implementation of the most important phenomena. Performing a somewhat more rigorous PIRT-like analysis would be highly advisable to provide a credible verification of the WindBlade code.

Table 1. HIGRAD solvers associated with the proposed test problems.

Test Problem	Conservation Laws	Flux Corrected Transport	Velocity Gradient	Hydrodynamic Discontinuity
Quiet Start	**X**			
Advection	**X**	**X**		
Diffusion	**X**		**X**	
Piston-like	**X**			**X**

The last column labeled “Hydrodynamic Discontinuity” refers to the ability of the fluid flow solver to capture interfaces, discontinuities, or mild “shocks.” Even though such flow conditions are not of interest for wind turbine applications, this phenomenology is nevertheless included to assess the extent to which HIGRAD is able to simulate these flow conditions. It makes for a particularly difficult test problem to “pass” because the code is not written to simulate these conditions. The code does not, for example, include models of artificial viscosity that are needed to “damp out” the spurious waves, known as Gibbs oscillation, generated at the flow discontinuity. The piston-like test problem simulated for this assessment is detailed in Appendix B.

Another consideration is the time devoted to the analysis of each individual problem. Running a test problem once to compare the discrete and exact solutions is not a concern, but performing a mesh refinement study rapidly becomes computationally expensive. Assuming, for example, that an initial grid is halved, then, halved again, the size of a three-dimensional problem grows by factors of eight, then, sixty-four. Solving the problem on the finest grid becomes 4,000 times more expensive than solving the problem on the initial grid if this resolution involves a solver whose computational cost is proportional to the square of the number of zones in the domain.

To alleviate this difficulty, it is decided to analyze the test problems of Table 1 on a rectangular, three-dimensional geometry. For the first three problems, the size of the domain is 64 x 64 x 64 m^3, whereas the domain analyzed for the piston-like problem is 0.5 x 10 x 0.5 cm^3. Keeping these domains rectangular greatly facilitates the meshing strategy and successive refinements. The fact that "real" geometries with, for example, realistic descriptions of the terrain or boundary conditions, are not used for the assessment is irrelevant. Because the object of verification is to assess the quality of algorithms implemented in the code, what matters is that the simplified test problems exercise the ***same*** models, algorithms, or lines of coding, as those that would be exercised in the application of interest. For better credibility, it would be highly advisable to augment this preliminary study with mesh refinement studies that involve realistic descriptions of the geometry, wind turbine placements, and terrain and atmospheric conditions.

Lastly, to ensure the stability of discrete solutions, it is important to guarantee that information is not propagated through the computational domain faster than the maximum wave speed ω_{Max} of the phenomenon simulated. Because the mesh size Δx used for a test problem is controlled by the domain size and other constraints imposed by the mesh refinement, the Courant-Friedrichs-Levy (CFL) stability condition is enforced by selecting an appropriate time step Δt such that:

$$\mathrm{CFL} = \frac{\omega_{\mathrm{Max}} \cdot \Delta t}{\Delta x} \leq 1 . \tag{10}$$

The CFL condition of equation (10) is obtained for each test problem using the maximum speed of sound expected to be encountered in the simulation. When mesh refinement is carried out, a common time sampling, therefore, based on the smallest level of resolution, is used. Unless otherwise noted in sections 4 to 7, the simulations are performed with Δt = constant such that no inconsistency is introduced in the discrete solutions due to a varying time sampling period. This rationale is deemed appropriate for a preliminary study because hydrodynamics solvers are generally written such that the temporal discretization is "slaved" to the spatial discretization through the CFL stability, and other, conditions. Hence, spatial truncation is the main production of numerical error. To verify the accuracy of the time integration scheme, the assessment of the scalar advection test problem in section 5 includes refinement studies in time as well as space.

3.4 Comparison Between Exact and Discrete Solutions

Code verification refers to the comparison of discrete approximations, obtained by simulating a test problem, to known exact solutions. Metrics, such as the absolute difference between scalar quantities or Euclidean distances estimated over the computational domain, can be calculated to assess the error between discrete and analytical solutions. One such metric used in this work is the Root Mean Square (RMS) error written, for example, as:

$$\varepsilon(\Delta x) = \sqrt{\frac{1}{N_K} \sum_{k=1}^{N_K} \left(y_k^{\mathrm{Exact}} - y_k^n \right)^2} , \tag{11}$$

where y_k^n is the numerical solution provided by HIGRAD at resolution Δx and y^{Exact} is the exact solution of the problem. Both solutions are discretized spatially over the same computational geometry, which is indicated in equation (11) by the summation over index k = 1 … N_K. (Note that a temporal error would be handled similarly by integrating over the time dimension instead of the spatial coordinates.)

The assessment, based on the metric $\varepsilon(\Delta x) = ||y^{\mathrm{Exact}} - y(\Delta x)||$, is straightforward: the code either passes or fails the test. Large errors indicate significant deviations from the exact solutions due

to programming mistakes. In general, estimating the error metric (11) will not indicate the origin of the discrepancy but a large error is a sure indicator that something is wrong.

3.5 Analysis of Code Performance With a Known Exact Solution

Another activity of code verification is to assess the numerical performance of an algorithm or solver implemented in the software. By repeating the analysis of the ***same*** test problem twice, but with different levels of mesh resolution, equation (11) can be calculated twice as:

$$\varepsilon\left(\Delta x_C\right)=\sqrt{\frac{1}{N_K^C}\sum_{k=1}^{N_K^C}\left(y_k^{Exact}-y_k^n(\Delta x_C)\right)^2}\,, \qquad \varepsilon\left(\Delta x_F\right)=\sqrt{\frac{1}{N_K^F}\sum_{k=1}^{N_K^F}\left(y_k^{Exact}-y_k^n(\Delta x_F)\right)^2}\,. \tag{12}$$

In equation (12), one of the two runs is referred to as the coarse-mesh calculation, while the other one is referred to as the fine-mesh calculation, to emphasize that different mesh sizes are used. The notation also introduces the symbols Δx_C and Δx_F that are characteristic mesh sizes of the coarse-mesh and fine-mesh calculations, respectively.

The definition of these characteristic mesh sizes can be problematic in multiple dimensions, and especially in the case of a Lagrangian mesh. Because the HIGRAD code is based on a Eulerian frame-of-reference, the definition of these characteristic sizes is trivial. In the problems analyzed next, the discretization is performed with a constant mesh size and unit aspect ratios.

Assuming that the truncation error behaves in the regime of asymptotic convergence according to the error Ansatz model of equation (9), and neglecting the higher-order terms, leads to:

$$\varepsilon\left(\Delta x_C\right) \;\approx\; \beta\,\Delta x_C^p\,, \qquad \varepsilon\left(\Delta x_F\right) \;\approx\; \beta\,\Delta x_F^p\,. \tag{13}$$

A solution for the pair of unknown parameters (p; β) is given by:

$$p=\frac{\log(\eta)}{\log(R)}, \tag{14}$$

and:

$$\beta=\frac{\varepsilon(\Delta x_C)}{\Delta x_C^p}=\frac{\varepsilon(\Delta x_F)}{\Delta x_F^p}, \tag{15}$$

where η is the ratio of coarse-to-fine solution errors and R is the ratio of coarse-to-fine sizes:

$$\eta=\frac{\varepsilon(\Delta x_C)}{\varepsilon(\Delta x_F)}, \qquad R=\frac{\Delta x_C}{\Delta x_F}\,. \tag{16}$$

The refinement ratio of equation (16) is, by definition, always greater than one since $\Delta x_C \geq \Delta x_F$. The procedure outlined in equations (12-16) indicates that two calculations with different levels of mesh resolution are sufficient to estimate the rate-of-convergence p. If additional runs are available, then a least-squares solver can easily be implemented to estimate the pair of parameters (p; β) that best-fits the model of truncation error of equation (9).

The observed rate-of-convergence estimated in equation (14) can be compared to the formal order of accuracy of the numerical method implemented in the code. Sub-optimal performance, for example, if p = 1.65 is observed when solving the test problem with a second-order accurate method, may be indicative of a mediocre implementation or, worst, a programming mistake.

Running code verification test problems is useful for two purposes. First, it assesses the performance of the code on specific test problems for which an exact solution y^{Exact} is known,

which allows code developers and analysts to make sure that no programming error is present. Second, it serves as benchmarking exercise to assess the rate-of-convergence of the numerical method implemented in the code. The main drawback, however, is that the exact solution of the continuous equations must be known. It seriously restricts the application of code verification to a few test problems that may not cover the full range of physics implemented in the code.

3.6 Self-convergence and Analysis of Code Performance Without an Exact Solution

We now turn our attention to the general case of solution verification, that is, the assessment of convergence of discrete solutions as the mesh size is refined, or $\Delta x \rightarrow 0$. The main difficulty is that, in general, the exact solution y^{Exact} of the continuous equations is unknown, which implies that the solution error $\varepsilon(\Delta x)$ as defined in equation (11) cannot be computed.

To circumvent this difficulty, three assumptions are made. The first assumption is to replace the exact solution y^{Exact} by a "reference" solution denoted by $y^{Reference}$. The commonly encountered practice of defining this reference as the solution of a highly refined run of the code is strongly discouraged because the observations that result are prone to interpretation errors. Instead, this reference is made another unknown of the analysis.

The second assumption is to specialize the analysis to scalar-valued quantities. These scalars are extracted from the calculation. They are local values, such as a peak pressure, or integrated over the entire computational domain, such as an average temperature. Further assuming that convergence is monotonic as the level of resolution is increased (this is the third assumption) transforms the standard Ansatz model of equation (9) into:

$$\varepsilon(\Delta x) \;=\; y^{\text{Reference}} - y_k^n \;=\; \beta\,\Delta x^p \;+ O\left(\Delta x^{p+1}\right). \tag{17}$$

It is emphasized that this simplified representation of truncation error is, as before, valid only within the regime of asymptotic behavior. The main difference is that equation (17) depends on a triplet of unknown parameters ($y^{Reference}$; p; β), hence, requiring a minimum of three code runs. The reader is referred to References [15] and [16] for detailed discussions of how to handle the case of non-monotonic convergence in a manner similar to the derivations that follow.

By repeating the analysis of the ***same*** test problem three times, but with different levels of mesh resolution, and neglecting the higher-order terms, equation (17) can be written three times as:

$$\begin{cases} y^{\text{Reference}} - y_k^n(\Delta x_C) \;=\; \beta\,\Delta x_C^p \\ y^{\text{Reference}} - y_k^n(\Delta x_M) \;=\; \beta\,\Delta x_M^p \\ y^{\text{Reference}} - y_k^n(\Delta x_F) \;=\; \beta\,\Delta x_F^p \end{cases}. \tag{18}$$

As before, the first run is referred to as the coarse-mesh calculation, performed at characteristic mesh size Δx_C. The second and third runs are referred to as the medium-mesh and fine-mesh calculations, performed at characteristic mesh sizes Δx_M and Δx_F, respectively.

The equations (18) are manipulated to arrive at the expression below that needs to be solved to calculate the unknown rate-of-convergence p:

$$p\;\log\left(R_{MF}\right) \;+\; \log\left(\frac{1-R_{CM}^p}{1-R_{MF}^p}\right) \;=\; \log\left(\eta\right), \tag{19}$$

where η is the ratio of solution differences and R_{CM} and R_{MF} are refinement ratios defined as:

$$\eta = \frac{y_k^n(\Delta x_M) - y_k^n(\Delta x_C)}{y_k^n(\Delta x_F) - y_k^n(\Delta x_M)}, \qquad R_{CM} = \frac{\Delta x_C}{\Delta x_M}, \qquad R_{MF} = \frac{\Delta x_M}{\Delta x_F}. \tag{20}$$

There is no closed-form solution to the nonlinear equation (19) when the refinement ratios R_{CM} and R_{MF} are different, and numerical optimization must be used to obtain the value of p that satisfies this expression. With a constant level of refinement across the three calculations, that is, $R_{CM} = R_{MF} = R$, the solution of equation (19) reduces to the well-known one:

$$p = \frac{\log(\eta)}{\log(R)}, \tag{21}$$

where η is the ratio of solution differences defined in equation (20).

To conclude the derivations, it can easily be verified that the reference solution is obtained as:

$$y^{Reference} = y_k^n(\Delta x_F) + \frac{y_k^n(\Delta x_F) - y_k^n(\Delta x_M)}{R^p - 1} = y_k^n(\Delta x_M) + \frac{y_k^n(\Delta x_M) - y_k^n(\Delta x_C)}{R^p - 1}, \tag{22}$$

while the pre-factor coefficient is obtained as:

$$\beta = \frac{y_k^n(\Delta x_C) - y_k^n(\Delta x_M)}{\Delta x_M^p \left(R^p - 1\right)} = \frac{y_k^n(\Delta x_M) - y_k^n(\Delta x_F)}{\Delta x_F^p \left(R^p - 1\right)}, \tag{23}$$

which completes the estimation of the triplet ($y^{Reference}$; p; β) of the solution error Ansatz model.

In the verification literature, equation (22) is recognized as the Richardson extrapolation [17]. It is often noted that the reference solution $y^{Reference}$ obtained in this manner is a better approximation of the exact-but-unknown solution y^{Exact} than the discrete solutions of the mesh refinement study $y_k^n(\Delta x_C)$, $y_k^n(\Delta x_M)$, and $y_k^n(\Delta x_F)$. This is referred to ***self-convergence*** because what is assessed is the ability of the code to self-converge to a reference solution as the level mesh refinement is increased. It is noted, however, that there is no guarantee that the extrapolated solution $y^{Reference}$ obtained in this manner will converge to the exact solution y^{Exact} as $\Delta x \rightarrow 0$. The numerical method implemented in the code may generate a systematic bias that will remain undiscovered as long as the exact solution is unknown.

4. The Quiet Start Test Problem

The quiet start test problem is used to assess the extent to which the code is able to remain stable, and avoids developing numerical "noise," over a significant number of time steps. Since HIGRAD is a non-hydrostatic model, this test ensures that initialization of a hydrostatic and neutrally stable background environmental state does not induce any perturbation flow. In essence this test simply verifies the balance between pressure gradient and buoyancy terms of the base state.

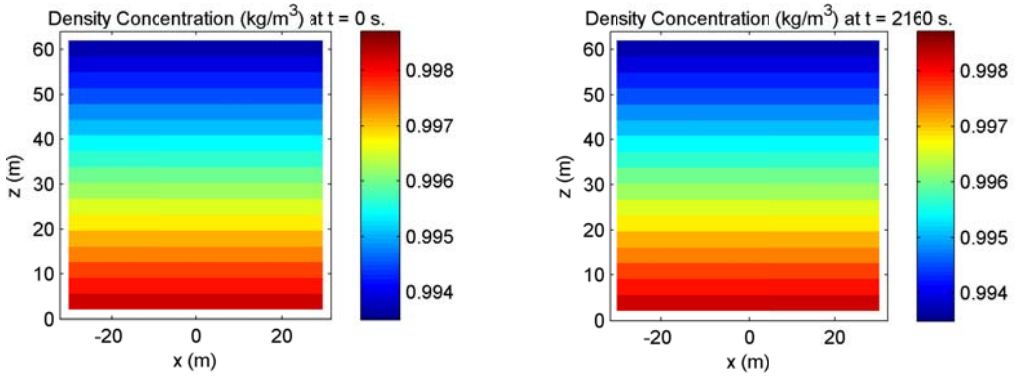

Figure 3-1. Coarse-size (C) solution of the quiet start problem.

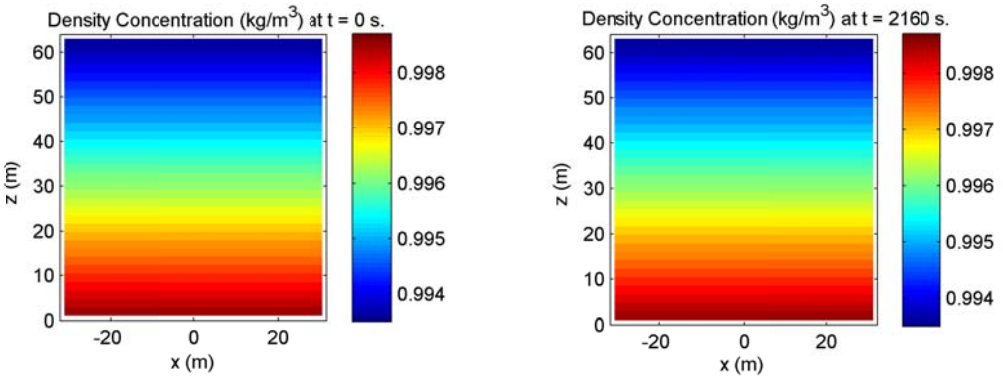

Figure 3-2. Medium-size (M) solution of the quiet start problem.

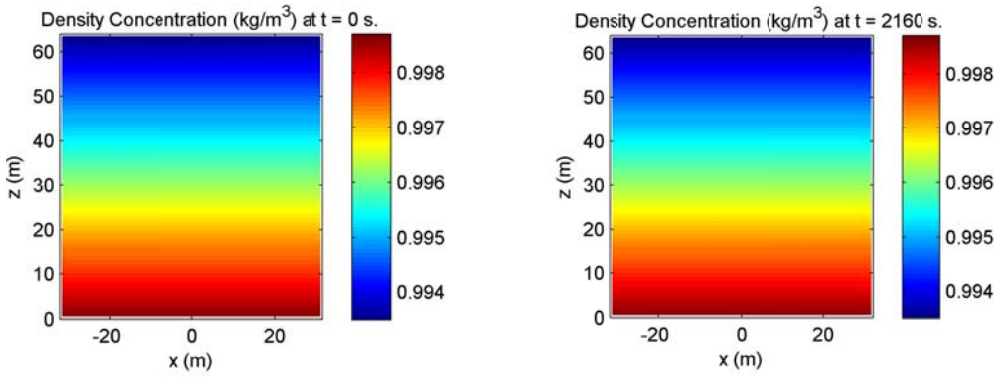

Figure 3-3. Fine-size (F) solution of the quiet start problem.

For this problem, the code is initiated at the initial time of $t = t_0$ and subsequent time steps are simulated. As the code is evaluated at $t > t_0$, the solution should remain equal to the initial condition defined at $t = t_0$. Three meshes are analyzed, first, to evaluate whether changing the

mesh generates a different level of "noise" and, second, to assess the numerical accuracy with a mesh refinement study. The domain is specified with an initial density concentration equal to $\rho_0 = 0.9987$ kg/m^3 at coordinate z = 0 m and $\rho_0 = 0.9935$ kg/m^3 at z = 64 m.

Figure 3 compares the discrete solutions calculated by the HIGRAD code at times t = 0 sec. and t = 2,160 sec. It is noted that a two-dimensional representation of these solutions may not be necessary. A time series of the resolved kinetic energy integrated over the computational domain should be constant, therefore, indicating that no flow is induced and the various terms that contribute to the conservation laws are balanced.

A visual comparison of left (at time t = 0 sec.) and right (at time t = 2,160 sec.) columns of the figure indicates that no disturbance is generated by the time integration. Similarly, the effect of increasing the level of spatial resolution is observed by comparing the coarse-size solution, medium-size solution, and fine-size solution. Again, no significant disturbance is generated.

Table 2. Root Mean Square (RMS) errors of the Oxygen density concentration.

Mesh Discretization	Time Sampling	Number of Cycles (Time Steps)	RMS Error of Solution $\|\|y^{Exact} - y(\Delta x)\|\|$
Coarse, Δx = 4 m	Δt = 0.20 sec.	N_{Cycle} = 43,200	0.000 kg/m^3
Medium, Δx = 2 m	Δt = 0.10 sec.	N_{Cycle} = 21,600	0.000 kg/m^3
Fine, Δx = 1 m	Δt = 0.05 sec.	N_{Cycle} = 10,800	0.000 kg/m^3

Table 2 is the counterpart of Figure 3 and it gives a quantitative comparison of the solution error obtained when analyzing the quiet start test problem with HIGRAD. The root mean square error $||y^{Exact} - y(\Delta x)||$ is calculated over the entire computational mesh as shown in equation (11). For each grid analyzed, the exact solution y^{Exact} is defined as the initial condition (at time t = 0 sec.) on that same mesh. This choice is made such that both exact and discrete solutions are defined on the same discretization, using zones that have identical volumes, which makes it easier to calculate the zone-to-zone differences.

The results of Table 2 suggest that the approximation provided by the simulation is "exact" even after performing a large number of time-integration cycles, and irrespective of the level of mesh resolution used in the calculation. Because the solution errors reported in the table are equal to zero, the rate-of-convergence of the numerical method cannot be estimated for this problem.

5. The Passive Scalar Advection Test Problem

The passive scalar advection test problem assesses the ability of the code to transport fluid due to bulk motion in a particular direction. Here, a bubble is prescribed at the center of the domain using an Oxygen concentration with an initial constant velocity of u_X = 6 m/s (meters per second) in the x-direction throughout the domain. (The positive x-axis points towards the right in the figures shown next.) Cyclic boundary conditions are defined, such that when the bubble reaches the edge of the domain at L_X = 64 m, it re-enters the domain on the left side at L_X = 0 m.

Simple physics, such as a dimensionless analysis of the mass continuity equation, can be used to calculate the time that it takes the bubble to move entirely through the computational domain and back to its starting position:

$$T = \frac{L_X}{u_X}, \tag{24}$$

where T denotes the time at which the solution is sought, L_X is the length of the domain, and u_X is the initial velocity. The exact solution is defined by translating the initial grid from its starting position to the location calculated at the time specified using equation (24). Figure 4 shows the exact solutions defined from the translated grids at t = 2.5 sec. (left) and t = 5.0 sec. (right).

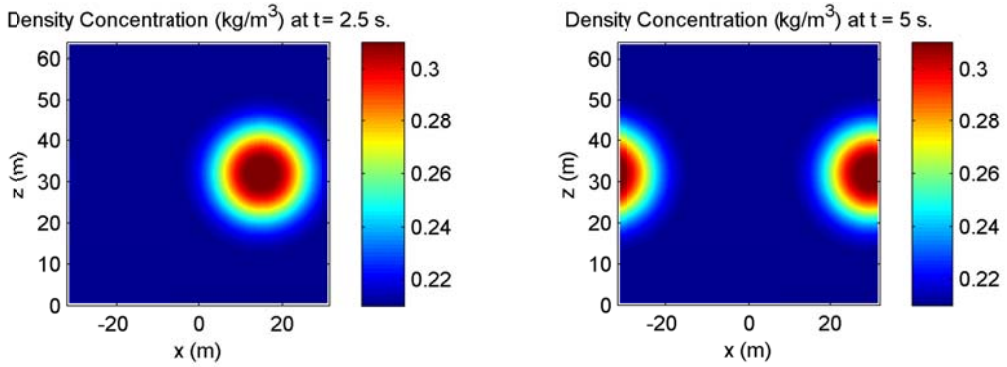

Figure 4. Exact solutions of the advection problem at t = 2.5 sec. and t = 5.0 sec.

The fact that, for this advection problem, the exact solution is simply a translation of the initial condition is because the initial state should move without any change through the computational domain since the boundary condition is cyclic and no source or "sink" term is added in the right-hand side of the governing equations. Any change of the solution computed by the code is a direct indication of "distortion" caused by the numerical differentiation of the advection operator. Numerical dissipation is one example of the mechanisms that could distort the initial condition.

Figure 5 gives a visual comparison of the numerical simulations at the same time of t = 5.0 sec. as the mesh is refined. For simplicity of implementation, the refinement is a grid halving, which means that the zones are divided in two in all directions between, for example, the medium-size and fine-size meshes. Even though only refinement in the direction of motion matters for this test problem, the aspect ratio is kept constant for all meshes analyzed. For the consistency of comparisons, the exact and discrete solutions are defined on the same grids. A fourth-order Runge-Kutta time-differencing scheme is specified for the numerical solver.

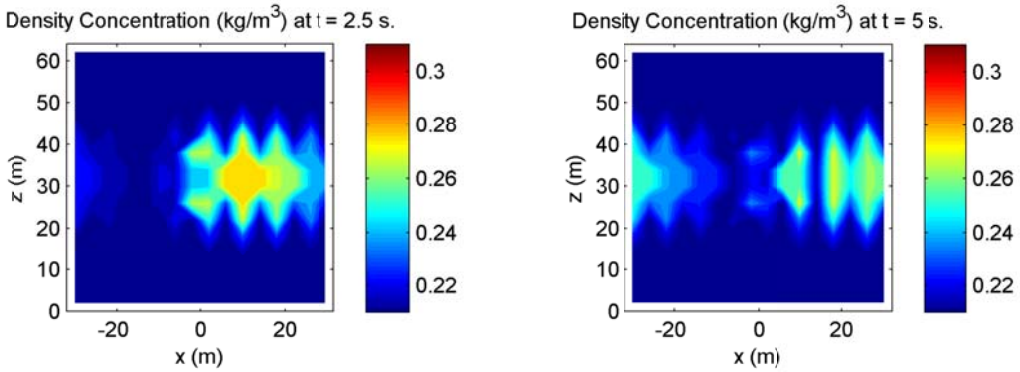

Figure 5-1. Coarse-size (C) solutions of the advection problem at 2.5 and 5.0 sec.

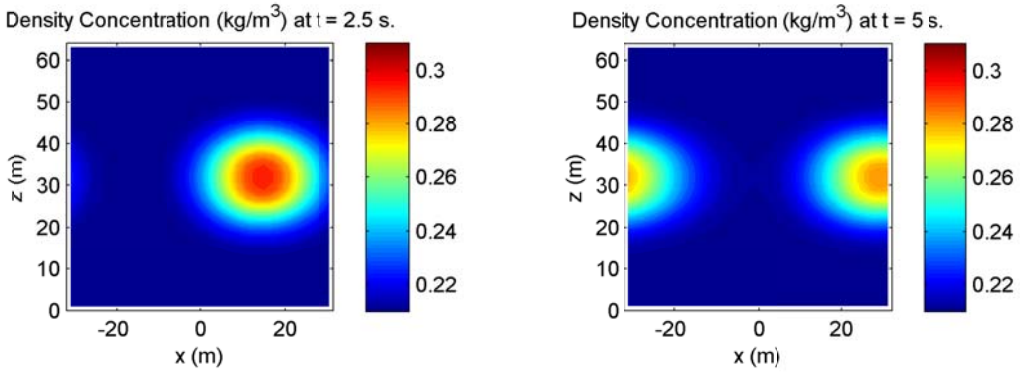

Figure 5-2. Medium-size (M) solutions of the advection problem at 2.5 and 5.0 sec.

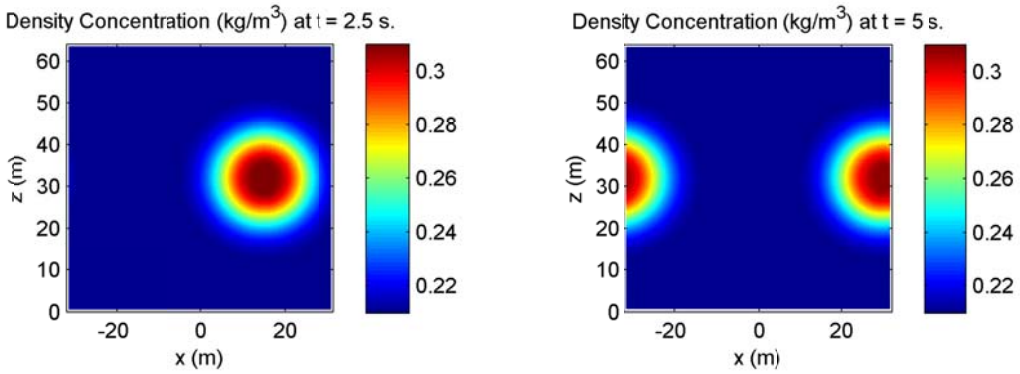

Figure 5-3. Fine-size (F) solutions of the advection problem at 2.5 and 5.0 sec.

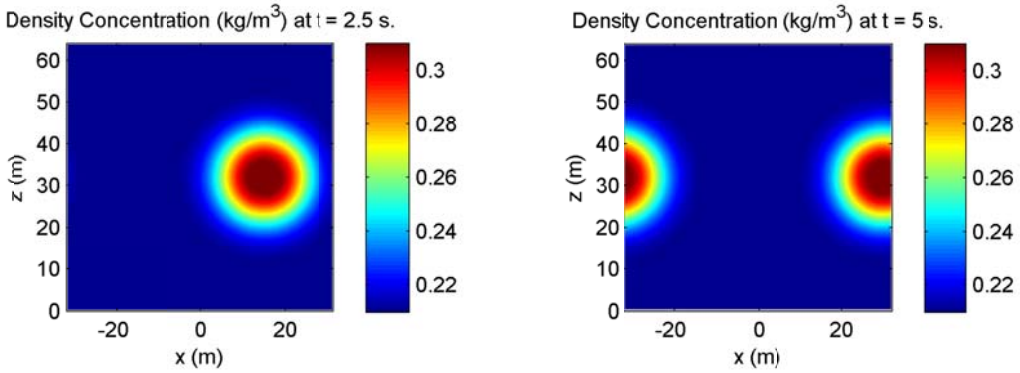

Figure 5-4. Extra Fine-size (XF) solutions of the advection problem at 2.5 and 5.0 sec.

Figure 5 indicates, in a "viewgraph norm" sense, the ability of the HIGRAD code to converge to the exact solution, shown in Figure 4, as the level of resolution in the calculation is increased. The discrete solution obtained with the coarsest level of spatial resolution clearly differs from the other ones. This is an example of calculation that is not even located in the regime of asymptotic convergence and, therefore, whose solution should be disregarded for the purpose of analyzing the performance of the code.

The coarse-size solution is obtained with a mesh size of $\Delta x = 4$ m, which corresponds to only 16 zones across the computational domain. This is not enough to sufficiently resolve the density gradient that defines the contour of the bubble. Careful examination of Figure 5-1 indicates the presence of eight "lobes." That the spatial gradient is spread over a number of components that happens to be exactly half the number of zones is a sure indication of spatial ***aliasing***.[1]

Table 3. Root Mean Square (RMS) errors of the Oxygen density concentration.

Mesh Discretization	RMS Error of Solution, $\|\|y^{Exact} - y(\Delta x)\|\|$		
	CFL = 0.300	**CFL = 0.150**	**CFL = 0.075**
Medium, Δx = 2 m	7.83×10^{-3} kg/m^3	5.51×10^{-3} kg/m^3	3.92×10^{-3} kg/m^3
Fine, Δx = 1 m	1.50×10^{-3} kg/m^3	0.87×10^{-3} kg/m^3	0.52×10^{-3} kg/m^3
Extra-fine, Δx = ½ m	0.25×10^{-3} kg/m^3	0.29×10^{-3} kg/m^3	0.22×10^{-3} kg/m^3

Table 3 lists the L^2 norms of differences between exact and discrete solutions, or RMS errors of equation (11), for the three finest levels of mesh resolution. The state variable analyzed is the density field in units of kg/m^3. Solutions are analyzed at the final time of t = 10.667 sec., when the Oxygen bubble initially located at the center of the domain has gone through one complete revolution. The exact solution is, therefore, defined as the initial condition at t = 0 sec.

Each column of Table 3 lists results for a constant CFL condition. The overall trend observed is a clear reduction of solution error as the grid is refined. Figure 6 is a graphical illustration of the data listed in the table, to which results obtained with other constant CFL conditions are added.

The RMS errors plotted in Figure 6 can be examined in two ways. At constant CFL number, one can examine the effect of increasing the mesh resolution. (More highly resolved calculations are towards the right.) At constant grid, one can examine the effect of decreasing the CFL condition, which translated into more temporal resolution as CFL $\rightarrow$ 0. In both cases, refining the model increases the computational demand but it also provides greater numerical accuracy.

In Figure 6 the RMS error is plotted on a linear scale, which is not conducive to estimating the order of accuracy, or rate at which the solution error is reduced. Analyzing results obtained with constant CFL values may also run the risk of compounding the effects of spatial and temporal resolutions in the calculation. This is because both mesh size Δx and sampling period Δt must be adjusted in order to keep the CFL number constant. With two parameters (Δx; Δt) varying

[1] The terminology "spatial aliasing" is used, here, in a broad sense; it is in analogy to the temporal aliasing of a waveform. In the discipline of signal processing, a high-frequency component at frequency f_K cannot be represented by sampling the signal at frequency Δf if f_K exceeds the Nyquist frequency. The Nyquist frequency is defined as $f_N = 2\Delta f$. If the condition $f_K \le f_N$ is violated, then the high-frequency component generates aliasing. It manifests itself by "folding" the high-frequency waveform over in the DC-to-f_N Hertz range and adding the appearance of a low-frequency component that, actually, does not exist. Because the temporal and spatial resolutions are related through the CFL condition, an upper limit "$f_K \le f_{Nyquist}$" in frequency translates to a lower limit "$\Delta x \ge \omega_{Max} \Delta t_N$ / CFL" in space. One uses equation (10) to translate one condition into the other one, together with the frequency-to-time conversion of $\Delta t_N = 1/f_N$.

simultaneously, one does not know whether the reduction in solution error is predominantly due to one effect versus the other one. For these reasons, a subset of the results is extracted from Figure 6 and presented in Figure 7 where time sampling is kept constant. Three sets of runs are shown for Δt = 50 milli-sec. (red), Δt = 25 milli-sec. (blue), and Δt = 12.5 milli-sec. (black).

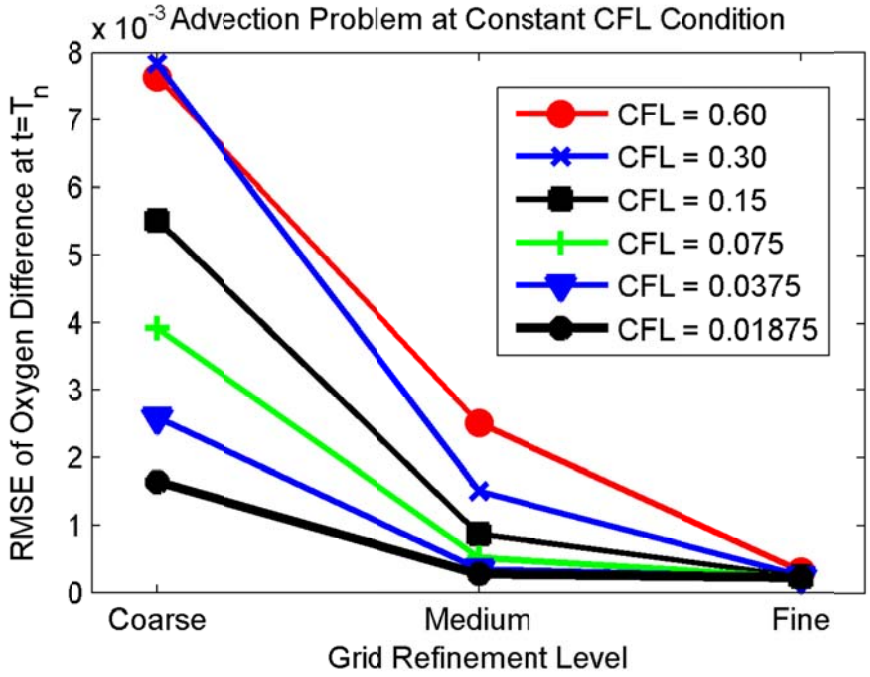

Figure 6. Solution errors for the advection problem, at constant CFL conditions.

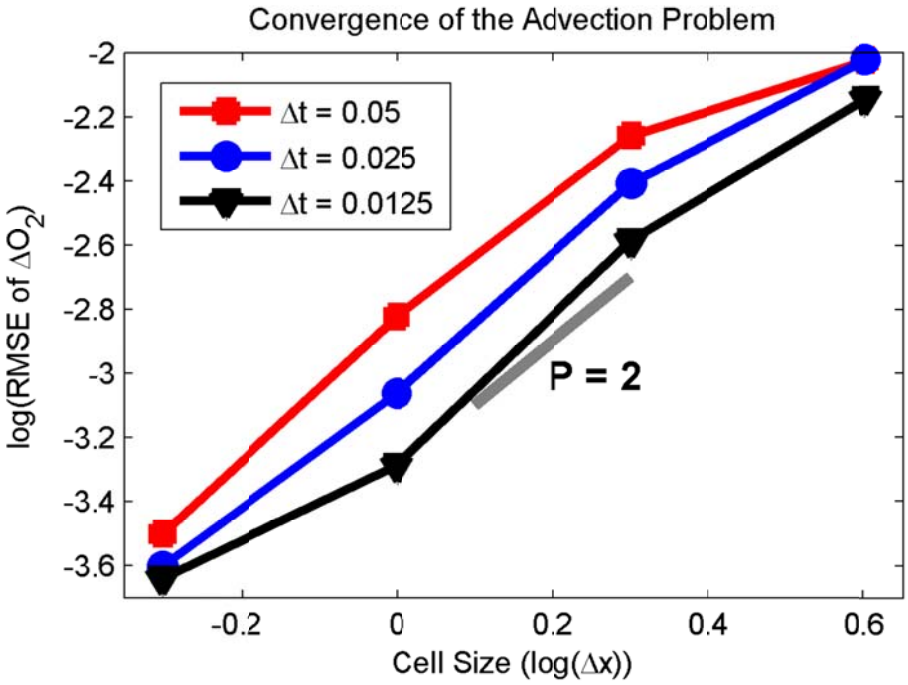

Figure 7. Solution errors for the advection problem, at constant time sampling.

The solution error is displayed in Figure 7 using a log-log scale. This is convenient to identify the rate-of-convergence of the numerical method. The figure clearly shows that the HIGRAD code displays steady monotonic convergence, irrespective of the time sampling period used, and that the order of accuracy is generally second-order. Table 4 quantifies these rates-of-convergence using the procedure of section 3.5 that requires data from only two computational grids.

Table 4. Estimation of the rates-of-convergence observed in Figure 7.

Pairs of Discrete Solutions Used	Rate-of-convergence of the Solution Error, p		
	Δt = 0.0125 sec.	**Δt = 0.025 sec.**	**Δt = 0.05 sec.**
Solution Pair (C; M)	p = 1.17	p = 1.80	p = 2.25
Solution Pair (M; F)	p = 2.33	p = 2.18	p = 1.87
Solution Pair (F; XF)	p = 1.47	p = 1.28	p = 0.78

Even though it is common practice, averaging the values listed in Table 4 to report a single, overall rate-of-convergence is not a valid procedure. A more correct approach is to best-fit, in a least squares sense, the solution errors to a parametric model such as $||y^{Exact} - y(\Delta x)||_2 = \beta\ \Delta x^p$ where the two unknowns are the pre-factor coefficient β and rate-of-convergence p. Figure 8 shows the results of this procedure for the four runs performed at Δt = constant = 25 milli-sec., which are shown in blue in Figure 7.

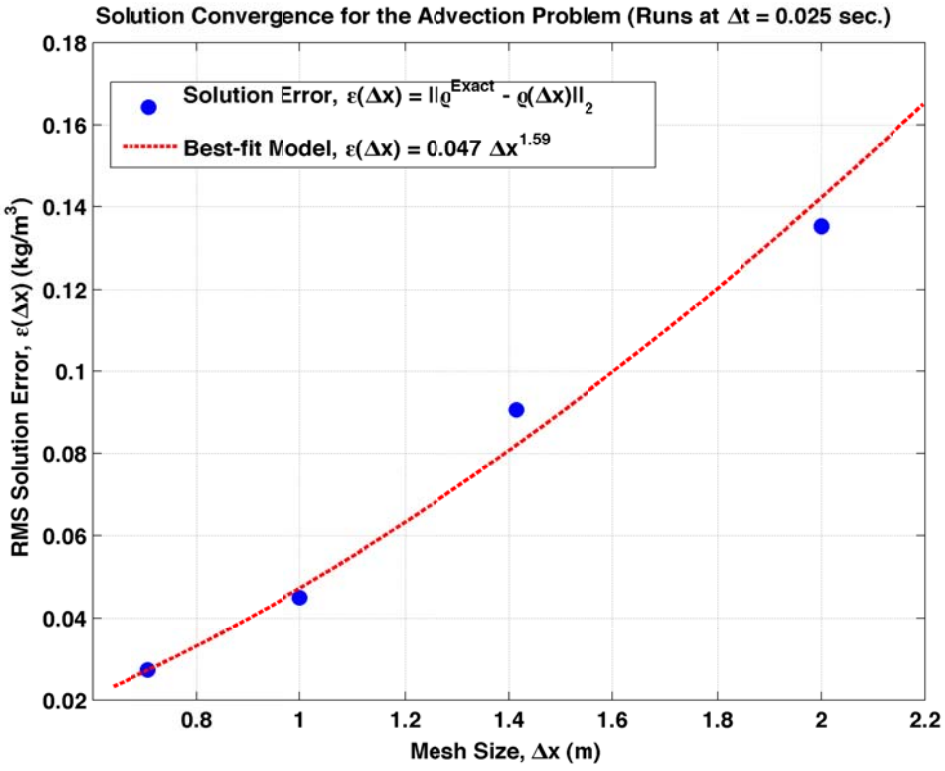

Figure 8. Best-fit of a global model $||y^{Exact} - y(\Delta x)||_2 = \beta\ \Delta x^p$ to runs at Δt = 25 milli-sec.

The results listed in Table 4 indicate that the observed rates-of-convergence vary in the range $0.78 \le p \le 2.33$, therefore, indicating an accuracy that tends to be better than first-order for the advection problem. This is an encouraging result because the formulation of the HIGRAD code

is generally second-order accurate. From Figure 8, where runs performed at Δt = 25 milli-sec. are analyzed, an overall rate-of-convergence of p = 1.59 is estimated. (The runs performed with other sampling periods exhibit similar properties.) This observation falls short of the expected second-order accuracy, which is not untypical of general-purpose hydrodynamics solvers and could be due to a number of factors.

The factor most likely responsible for deteriorating the accuracy of the method is the treatment of the boundary condition where discretization of the gradient operator may only be first-order accurate. Other factors, although unlikely for this problem, include round-off errors and improper definitions of variable types (such as real values stored as integers). It is noted, however, that the overall levels of solution error $||y^{Exact} - y(\Delta x)||$ remain low relative to absolute values of the state variables. Not matching exactly the theoretical order of accuracy implies few detrimental consequences and our view is that it does not "fail" the code verification exercise. The most significant concern is that a sub-optimal accuracy tends to increase the numerical uncertainty due to truncation effects. This may matter for activities such as test-analysis correlation where a prediction and its bounds of numerical uncertainty are compared to a measurement and its bounds of experimental variability, but it is not critical here.

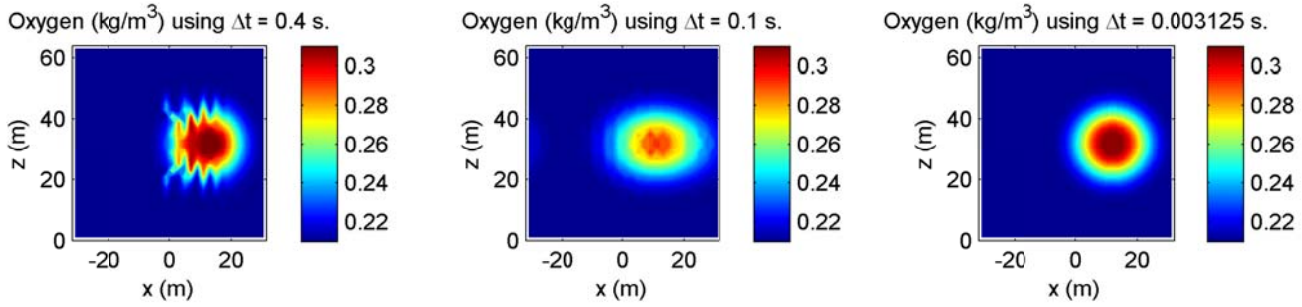

Figure 9-1. Time refinement of the advection problem using 1st-order Runge Kutta.

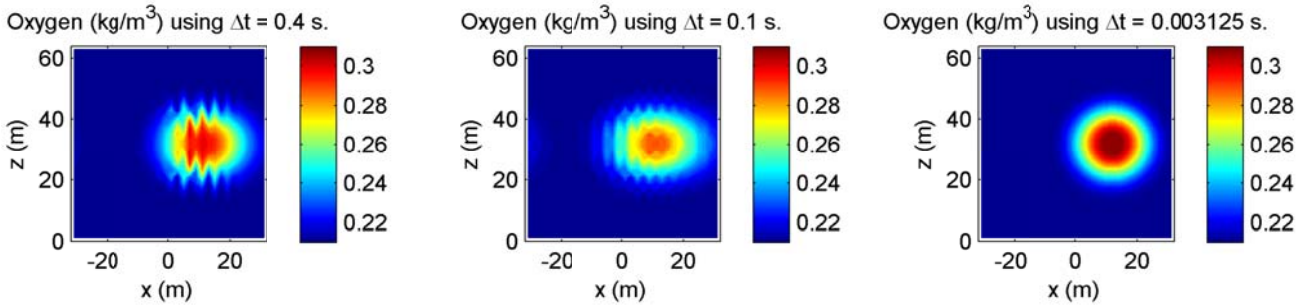

Figure 9-2. Time refinement of the advection problem using 3rd-order Runge Kutta.

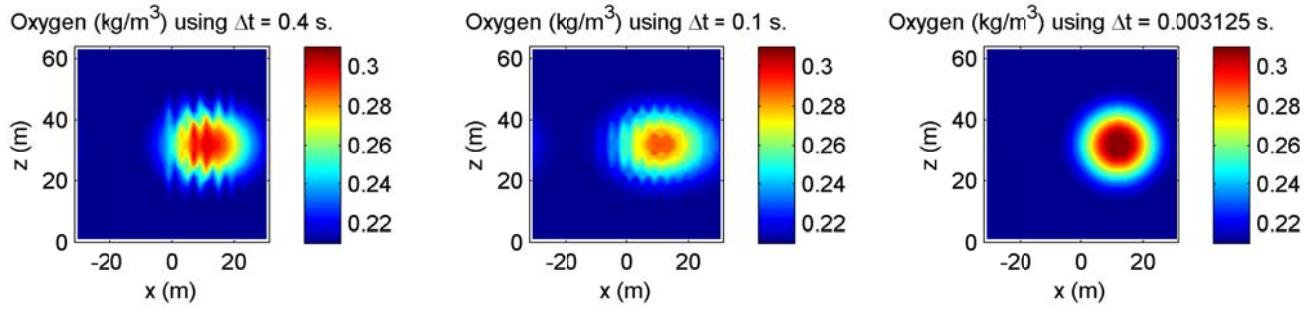

Figure 9-3. Time refinement of the advection problem using 4th-order Runge Kutta.

Next, the ability of the code to self-converge as a function of time resolution is assessed. The mesh size is held constant at $\Delta x = 2$ m, which is the previously labeled "coarse-size" mesh with 32 zones along the x-axis. Runs of the advection test problem are repeated by varying the time sampling period Δt. Furthermore, three time-differencing schemes are exercised: the 1st-order Runge-Kutta (RK-1), 3rd-order Runge-Kutta (RK-3), and 4th-order Runge-Kutta (RK-4) methods. Figure 9 shows a visual representation of how the code behaves when (i) the time step is refined to a smaller sampling (from left to right) and (ii) a higher-order time-differencing scheme is utilized (from top to bottom).

Since the fluid model is compressible, the fastest characteristic velocity, which is denoted by the symbol ω_{Max} in the CFL condition (10), is the speed-of-sound. Using the sound speed in the CFL condition causes some of the cases analyzed to be unstable. Figure 9 supports this claim by showing that the density fields predicted with coarser time resolutions exhibit the signature of an unstable CFL condition. Specifically, these are the runs performed with $\Delta t = 400$ milli-sec. and shown in the left-most column of Figure 9, and $\Delta t = 100$ milli-sec. in the middle column. Given a sound speed of $\omega_{Max} = 343$ m/sec. in ambient-temperature air, approximately, obtaining a stable CFL condition requires a sampling period no greater than $\Delta t \leq 0.012$ sec. (or 12 milli-sec.), which is a condition met by the runs shown in the right-most column with $\Delta t = 3.125$ milli-sec.

Table 5. RMS errors of Oxygen density concentrations for the time refinement study.

Space and Time Discretization Parameters		Density RMS Error (x 10^{-3} kg/m^3)		
Mesh Size	**Time Step, Δt (milli-sec.)**	**RK-1**	**RK-3**	**RK-4**
Coarse (Δx = 2 m)	**50.0**	1.742	1.935	1.935
Coarse (Δx = 2 m)	**25.0**	0.963	1.087	1.087
Coarse (Δx = 2 m)	**12.5**	0.507	0.577	0.577
Coarse (Δx = 2 m)	**6.25**	0.255	0.293	0.293
Coarse (Δx = 2 m)	**3.125**	0.121	0.140	0.140

The results in Figure 9 can be visually compared to the reference solution of Figure 4. As the time step is refined and a higher-order time-differencing scheme is utilized, the code produces predictions that better match the exact solution. The solution error is calculated using the RMS metric of equation (11) and the exact solution defined on the coarse-size mesh with $\Delta x = 2$ m. The RMS errors are listed in Table 5 and illustrated in Figure 10.

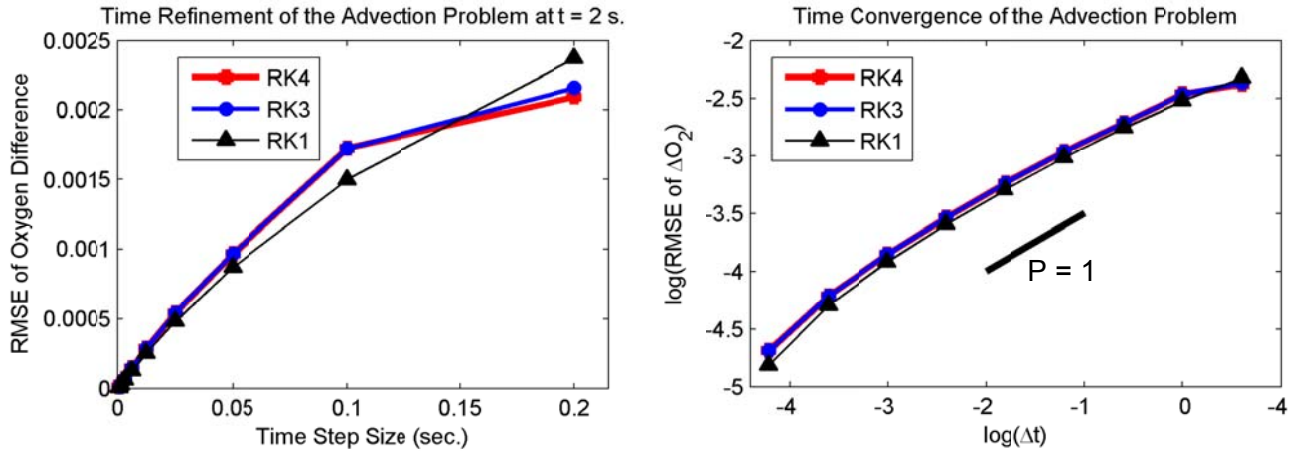

Figure 10. Solution errors for the time refinement study of the advection problem.

Table 6. Estimation of the rates-of-convergence observed in Figure 10.

Pairs (Δt_C; Δt_F) of Density Solutions Used	Rate-of-convergence of the Solution Error, p		
	RK-1	RK-3	RK-4
(Δt_C = 200.0; Δt_F = 100.0)	p = 0.66	p = 0.33	p = 0.28
(Δt_C = 100.0; Δt_F = 50.0)	p = 0.78	p = 0.83	p = 0.83
(Δt_C = 50.0; Δt_F = 25.0)	p = 0.85	p = 0.83	p = 0.83
(Δt_C = 25.0; Δt_F = 12.5)	p = 0.93	p = 0.91	p = 0.91
(Δt_C = 12.5; Δt_F = 6.25)	p = 0.99	p = 0.98	p = 0.98
(Δt_C = 6.25; Δt_F = 3.125)	p = 1.08	p = 1.06	p = 1.06
(Δt_C = 3.125; Δt_F = 1.5625)	p = 1.25	p = 1.20	p = 1.20
(Δt_C = 1.5625; Δt_F = 0.78125)	p = 1.73	p = 1.58	p = 1.58

Figure 10 shows the solution error $||y^{Exact} - y(\Delta t)||$ versus the time sampling period Δt on a linear scale (left side) and log-log-scale (right side). The log-log scale allows to examine visually the rate-of-convergence in time. The expectation is that the observed rate-of-convergence should correspond to the theoretical order of accuracy for each Runge-Kutta time integration scheme. The RK-4 method, for example, should be 4^{th}-order accurate. The three schemes analyzed, however, demonstrate nearly 1^{st}-order accuracy, without much gain in performance as a higher-order method is exercised.

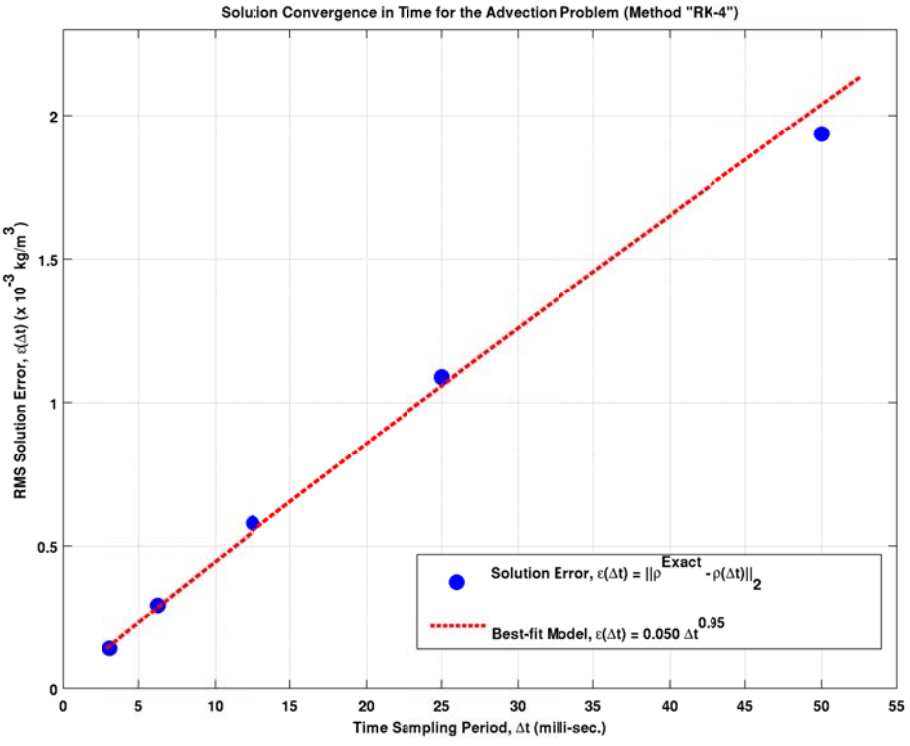

Figure 11. Best-fit of a global model $||y^{Exact} - y(\Delta t)||_2 = \beta\ \Delta t^p$ for time scheme RK-4.

Table 6 quantifies the rates-of-convergence in time by applying the procedure of section 3.5 that requires data from only two runs. The accuracy is sub-optimal, as expected, when the runs used in the analysis violate the CFL stability condition. This is the case, for example, on the first row of the table where the solutions obtained with 200.0 and 100.0 milli-sec. are used. The results tabulated indicate that the convergence rates then stabilize at exactly first-order accuracy before improving somewhat when the runs are performed with very small time sampling. For reference, analyzing the advection problem with a sampling period of $\Delta t = 0.78125$ milli-sec. corresponds to a stability condition of CFL = 0.067.

Figure 11 is a counterpart to Figure 8 where a global model $||y^{Exact} - y(\Delta t)||_2 = \beta\, \Delta t^p$ is best-fitted to the solution errors obtained for the 4^{th}-order Runge-Kutta time integration scheme, which is the right-most column in Table 5. It gives an overall rate-of-convergence of $p = 0.95$ without having to rely on the procedure outlined in section 3.5. The other integration methods (RK-1 and RK-3) behave similarly.

The fact that the formal order of accuracy in time is not recovered for the higher-order methods may be due to the stability demand of the CFL condition. This makes it difficult to “isolate” the truncation effects due to temporal discretization from those due to spatial discretization. Based on the observations made here, the time convergence study is not replicated with the other test problems. The sub-optimal performance of time integration remains, at this point, unexplained and is a potential area of re-investigation.

6. The Passive Scalar Diffusion Test Problem

The third test problem analyzes the simulation of passive diffusion. Physically, diffusion is the process by which particles spread from regions of higher concentration to regions of lower concentration. The analytical problem tested in the HIGRAD code starts with a domain specified with a constant initial Oxygen concentration, q_∞ = 0.22 kg/m^3 throughout. A constant Oxygen density, Q = 0.32 kg/m^3 is diffused into the domain at a constant rate from the top boundary. As a result, the Oxygen concentration within the domain increases, starting at the top boundary and eventually reaching the bottom boundary. If the diffusion process is allowed to continue forever, that is, until t → +∞, then the Oxygen concentration of the domain would converge to Q.

The problem setup uses an essential boundary condition at the top and bottom of the domain, where L_Z = 0 m, and L_Z = 64 m, respectively. The side edges of the domain are defined with natural boundary conditions, where the time rate of change of the Oxygen concentration is zero. Fick's second law of diffusivity can be used to predict how the concentration changes with time. Details of the derivation of the closed-form analytical solution are provided in Appendix A, with the resulting solution given as a function of time as:

$$q = q_\infty + \sum_{l=1}^{\infty} \frac{4(Q-q_\infty)}{(2l-1)\pi}(-1)^{l-1} \cdot e^{-\kappa\tau_l t} \cdot \cos\left(\frac{(2l-1)\pi z}{2L_z}\right). \tag{25}$$

Figure 12 illustrates the closed-form solution at t = 240 sec. (middle) and t = 480 sec. (right), obtained with q_∞ = 0.22 kg/m^3 and Q = 0.32 kg/m^3. The left-most figure shows the uniform initial condition at q_∞ = 0.22 kg/m^3. The figure shows that as time increases, the Oxygen concentration in the domain increases, first, near the top of the domain. It, then, progressively approaches the bottom of the domain. Even though the analytical solution is one-dimensional along the z-axis, the test problem is analyzed in three-dimensional, Cartesian geometry to verify that the discrete solutions remain one-dimensional and that this essential symmetry is not lost.

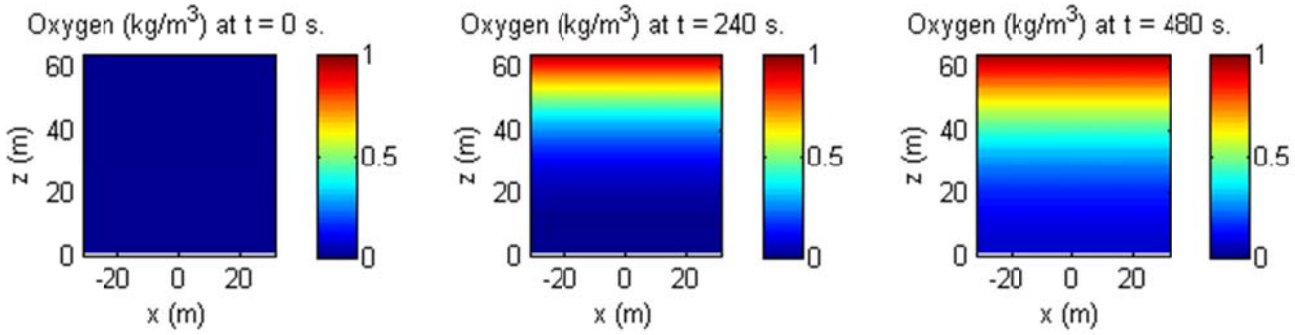

Figure 12. Exact solutions of the diffusion problem at t = 240 and t = 480 sec.

Equation (26) indicates that the closed-form solution decays exponentially from top to bottom for times t ≥ 0. This is visible in Figure 12 where the view shown is the entire domain extending vertically between coordinates 0 m ≤ z ≤ 64 m.

A mesh refinement study is performed to, simultaneously, calculate the solution RMS error and estimate the rate-of-convergence for the diffusion problem. The exact and discrete solutions are compared in Figure 13. The left-most column of Figure 13 shows the discrete solutions obtained by running the HIGRAD code with different levels of mesh resolution. The right-most column is a comparison to the exact solution (26) evaluated on the same computational grid.

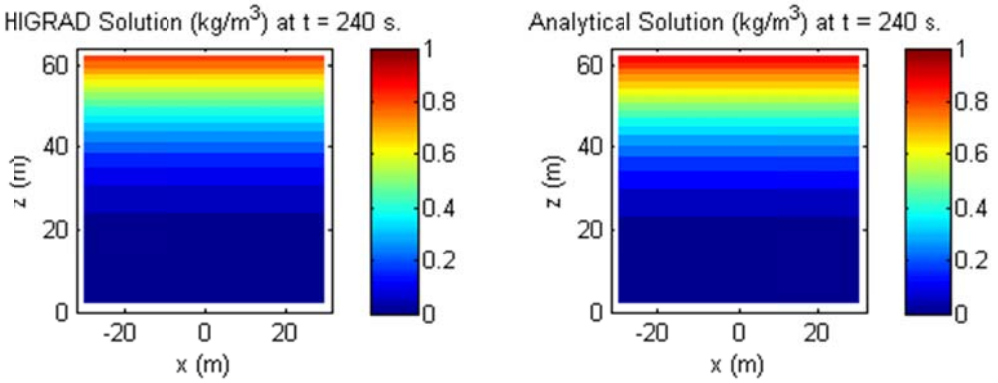

Figure 13-1. Coarse-size (C) Oxygen concentration of the diffusion problem.

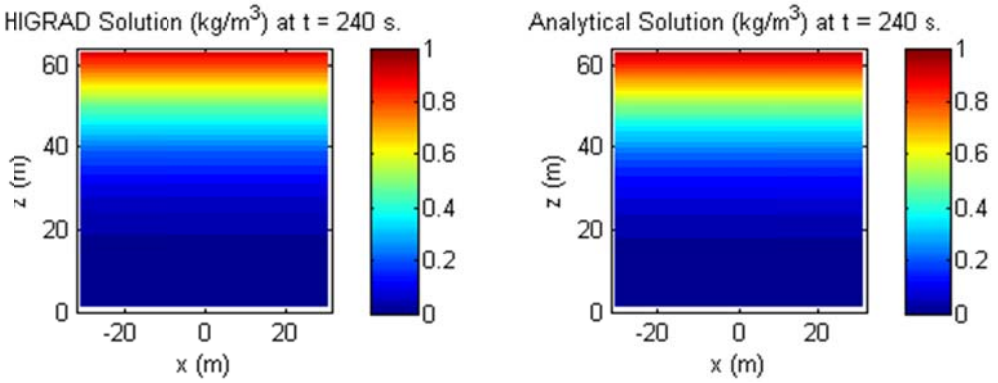

Figure 13-2. Medium-size (M) Oxygen concentration of the diffusion problem.

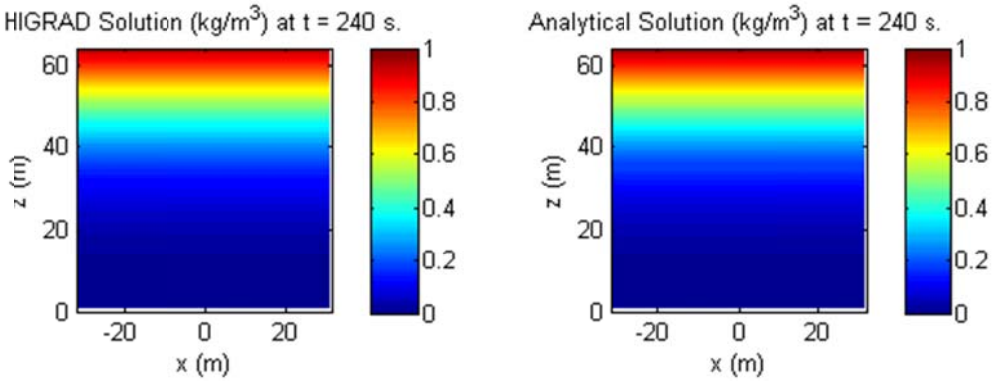

Figure 13-3. Fine-size (F) Oxygen concentration of the diffusion problem.

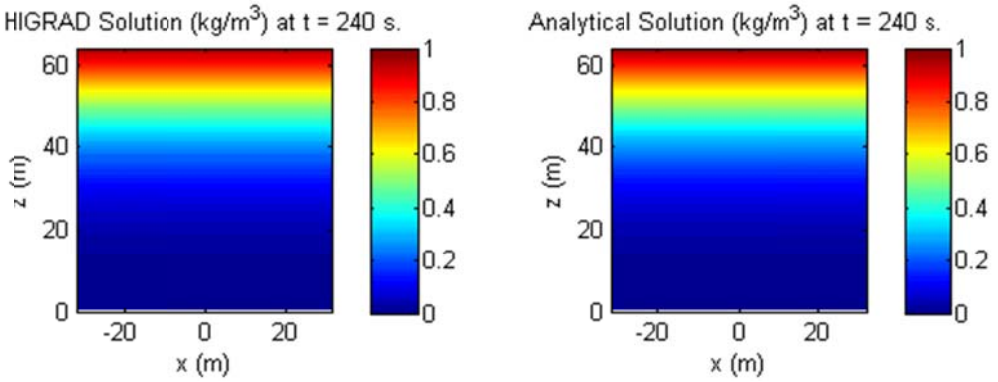

Figure 13-4. Extra fine-size (XF) Oxygen concentration of the diffusion problem.

As discussed previously for the other test problems, the analytical solution is re-generated for each mesh size. This is important to enforce the comparison of exact and discrete solutions defined in the same mathematical space. The view shown in Figure 13 is of the diffusion transition region comprised between coordinates 0 m ≤ z ≤ 64 m. It is clear from this "viewgraph norm" that, as the mesh is refined, the simulation is more capable of replicating the closed-form solution. Equally important is the fact that the solution symmetry is preserved.

The RMS norms of zone-to-zone differences between the exact and discrete solutions are given in Table 7. Two cases are analyzed. The first case applies to solution differences defined over the entire computational domain, which is the cube of size 0 m ≤ L_X, L_Y, L_Z ≤ 64 m. The second case applies to an interior region defined by the cube of size 16 m ≤ L_X, L_Y, L_Z ≤ 48 m. In other words, the second case is an analysis of a sub-region distant of 16 m from the edges of the domain. The inner cube is evaluated to assess if the treatment of the boundary condition affects the solution error and observed order of accuracy.

Table 7. RMS errors of Oxygen density concentrations for the diffusion problem.

Mesh Used	Mesh Size	RMS Errors (x 10^{-3} kg/m^3) in the Inner Cube	RMS Errors (x 10^{-3} kg/m^3) in the Entire Domain
Coarse (C)	**Δx = 4 m**	9.01	39.06
Medium (M)	**Δx = 2 m**	2.24	19.79
Fine (F)	**Δx = 1 m**	0.56	9.97
Extra Fine (XF)	**Δx = ½ m**	0.14	4.99

The RMS errors of solution differences defined over the entire domain are listed in the second column of Table 7. Those of the interior cube are given in the third, and right-most, column. In both cases, the exact and discrete solutions are taken at t = 240 sec. The solution errors clearly indicate the influence of the boundary condition. Including zones located near the edges of the domain significantly increases the overall error.

In a relative sense, however, these differences between the exact and discrete solutions only represent small percentages of the overall solution magnitude. For example, the worst-case error reported in Table 7 for the entire domain, that is, $\varepsilon(\Delta x_C) = 39.06\ 10^{-3}$ kg/m^3 obtained with the coarse-size grid, gives 15% error. The best-case of $\varepsilon(\Delta x_{XF}) = 4.99\ 10^{-3}$ kg/m^3 obtained with the extra fine-size grid translates to 2% error. These low levels of error, relative to the solution magnitude, indicate that HIGRAD solves the linear diffusion equation in a satisfactory manner.

Figure 14 is a companion to Table 9 that illustrates graphically the behavior of solution error as a function of mesh size, and for different conditions of CFL and time step controls. The figure on the left plots the solution error ε(Δx) versus mesh size Δx for different CFL conditions. The figure on the right plots the solution error for different time steps. (The time sampling frequency is kept constant in all these simulations.) It can be observed that, in both cases, running the calculation at a different condition of CFL stability or time step controls has no effect on the error generated by the code. This is true whether the analysis is applied to the entire domain or restricted to the inner cube and away from the boundary condition.

The log-log scale used on the right side of Figure 14 makes it easy to observe the convergence of the numerical method. Here, using the discrete solutions defined over the entire domain leads to first-order accuracy, approximately. Restricting the solutions to the inner cube region yields an unambiguous second-order convergence. The linear diffusion analyzed in this test problem is represented by a Laplacian operator, which produces "smooth" and well-behaved solutions. The

expectation, therefore, is to observe the theoretical second-order accuracy. Figure 14 indicates that this is precisely what happens when zones located near the edges are eliminated.

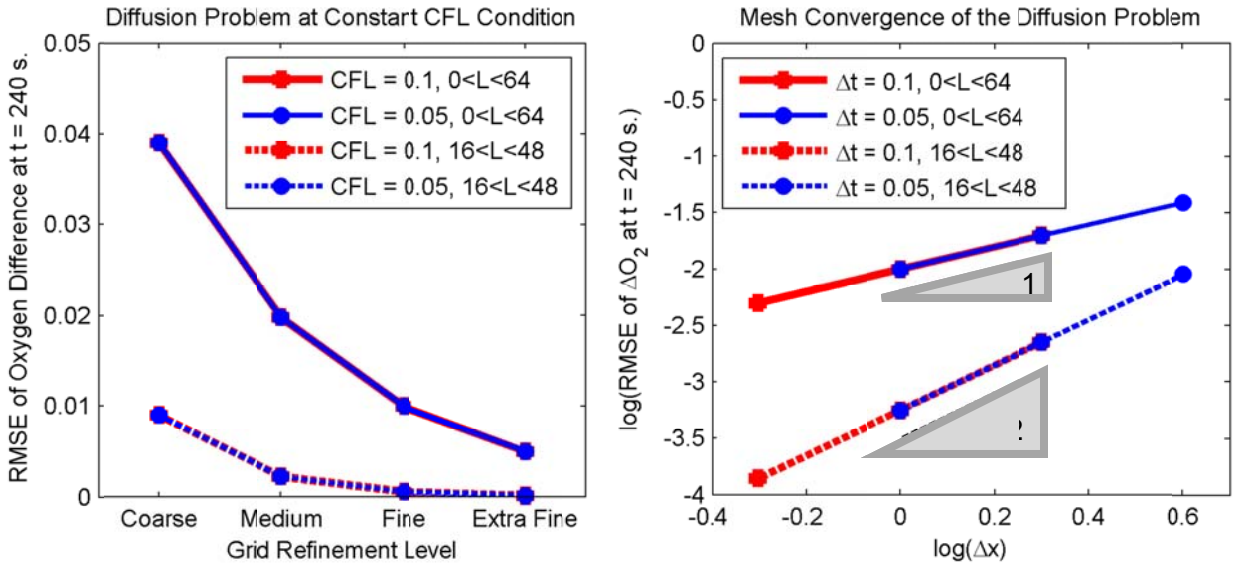

Figure 14. Solution errors for the diffusion problem, at different CFL/Δt conditions.

Table 8 lists the rates-of-convergence estimated using the procedure detailed in section 3.5. This quantitative analysis confirms that the 2nd-order accuracy of solutions observed inside the domain deteriorates to 1st-order when regions near the edges are considered for analysis. The rates-of-convergence obtained are very consistent and, as expected, insensitive to variations of the CFL and time step controls. Note that, when the convergence rates estimated with different pairs of solutions turn out to be so consistent, a global estimation such as the one of section 5 is not necessary. Best-fitting a single model of truncation error, $||y^{Exact} - y(\Delta x)||_2 = \beta\ \Delta x^p$, to the multiple runs available yields identical results.

Table 8. Estimation of the rates-of-convergence observed in Figure 14.

Pairs Used	Using Errors in the Inner Cube		Using RMS Errors in the Entire Domain	
	Δt = 0.05 sec.	**Δt = 0.10 sec.**	**Δt = 0.05 sec.**	**Δt = 0.10 sec.**
(C; M)	p = 2.00	N/A	p = 0.99	N/A
(M; F)	p = 2.01	p = 2.01	p = 0.99	p = 0.99
(F; XF)	N/A	p = 2.01	N/A	p = 0.98

The rates-of-convergence of Table 8 confirm what is observed in Figure 14: the accuracy of the simulation is significantly deteriorated when zones located near the boundary of the domain are included in the analysis. Even though this has not been verified, the most likely explanation is that the numerical treatment of the boundary condition is responsible for reducing the accuracy.

To avoid propagating boundary-induced errors, a practical recommendation would be to mesh a larger-than-needed computational domain, which is trivial with a Eulerian method. Preventing the contamination of the solution, however, depends on the flow since errors tend to propagate alongside waveforms carried by the flow. An interesting follow-up study would be to assess how closely to the edges of the domain does the solution remain essentially 2nd-order accurate.

7. Variant of the Noh Test Problem

The last problem analyzed in this report is a variant of the well-known Noh problem.[18] The variant presented is derived to verify the fluid dynamics solver when simulating flow that feature "mild" discontinuities. Even though discontinuous flows, such as shock waves, are not relevant to the application of interest, this test problem is considered to push the limits and examine how the solver performs in a situation for which it was not designed. Because the flow conditions for which the equations are solved include those of compressible flows, it could be argued that it is appropriate to test the performance of HIGRAD in these conditions.

The mesh is initialized with a constant velocity field across the domain pointing towards the right side, that is, (u_X; u_Y; u_Z) = (+constant; 0; 0). The right boundary is a rigid wall where u_X = 0. The pressure and internal energy are initialized at zero within the domain. Dimensions of the domain are 10 x 0.5 x 0.5 cm^3.

Because of the presence of a rigid wall, a discontinuity in pressure and energy forms "instantaneously" at the boundary and starts to propagate towards the left side, that is, towards the negative X-values, as shown in Figure 15 at the times of 125 micro-sec. (middle) and 250 micro-sec. (right). The flow generates a "jump" condition that separates the undisturbed and post-shock regions. In the post-shock region, the pressure and density increase to constant values while the velocity drops to zero (no motion). This test problem, therefore, provides a verification of the ability of the solver to convert kinetic energy into internal energy.

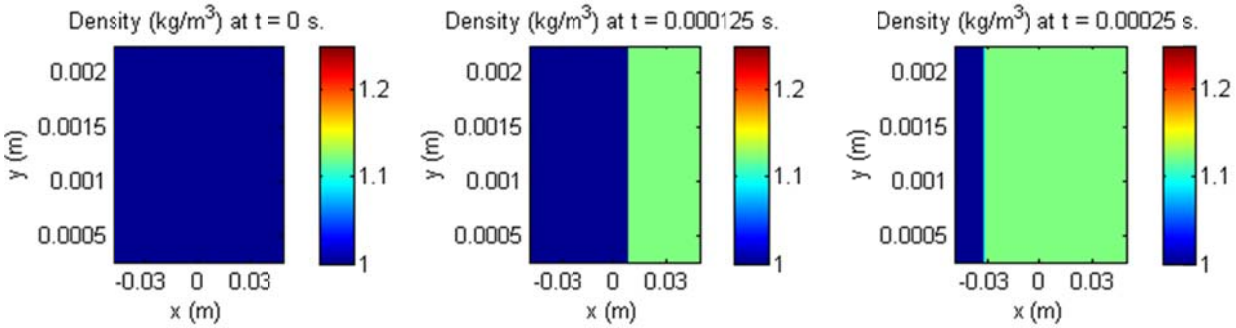

Figure 15. Exact solution for the Noh-like problem, at t = 0, 125, 250 micro-sec.

Derivations of the exact solution illustrated in Figure 15 are presented in Appendix B. The exact solution is adapted from the solution of the Noh problem to account for the fact that, because the conservation laws are solved in a Eulerian frame-of-reference, the boundary conditions are not exactly those of the conventional Noh problem. This introduces a perturbation handled by examining only what happens in the immediate vicinity of the "jump" condition and away from the edges of the domain.

As mentioned previously, the test problem tests two basic properties of a hydrodynamics code. Because motion comes to a halt behind the discontinuity, the problem assesses, first, the ability of the solver to convert kinetic energy into internal (or potential) energy. Second, it assesses the extent to which the solver captures properties of the discontinuity, also referred to as Rankine-Hugoniot "jump" conditions. Figure 16 compares the discrete solutions obtained with four grids, starting with 100 zones in the direction of wave propagation, that is, Δx = 1 mm. The aspect ratio of zones is kept constant and equal to one. Mesh halving is used to generate the other three meshes that, therefore, are analyzed with 200 zones (medium-size), 400 zones (fine-size), and 800 zones (extra fine-size) along the X-axis.

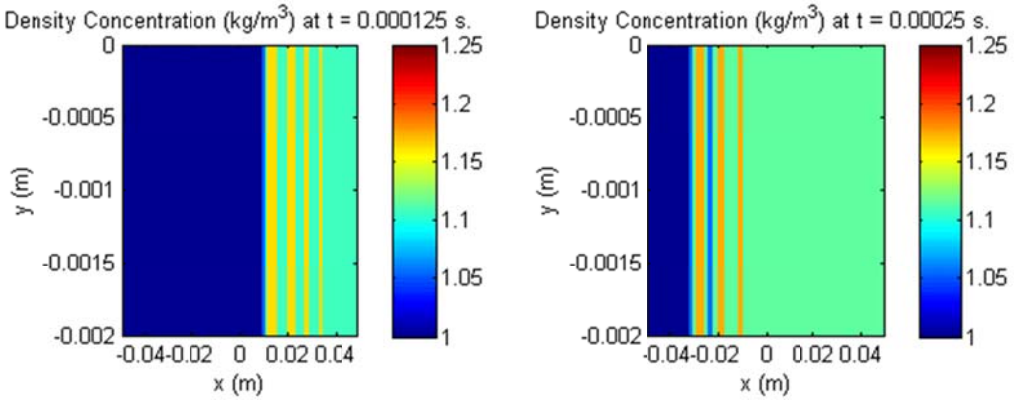

Figure 16-1. Coarse-size (C) density solution of the Noh-like problem.

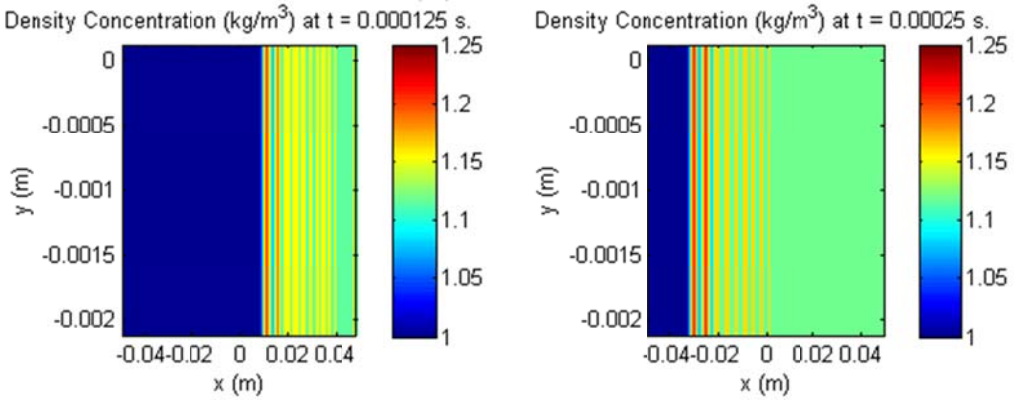

Figure 16-2. Medium-size (M) density solution of the Noh-like problem.

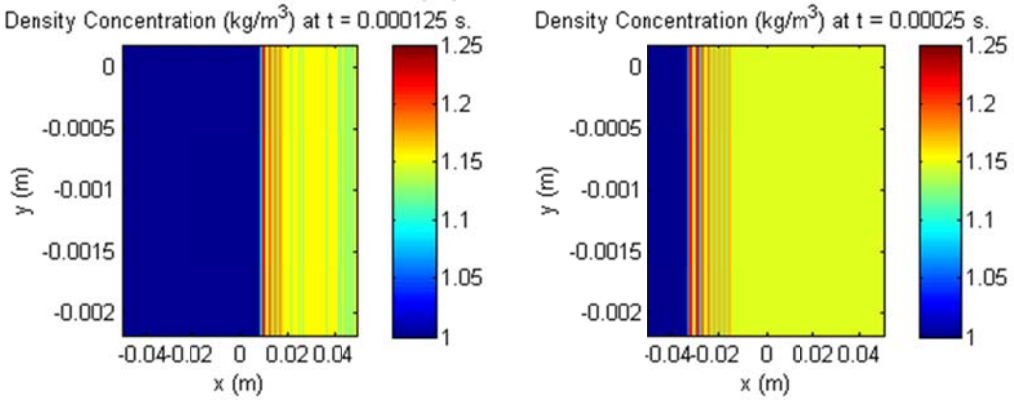

Figure 16-3. Fine-size (F) density solution of the Noh-like problem.

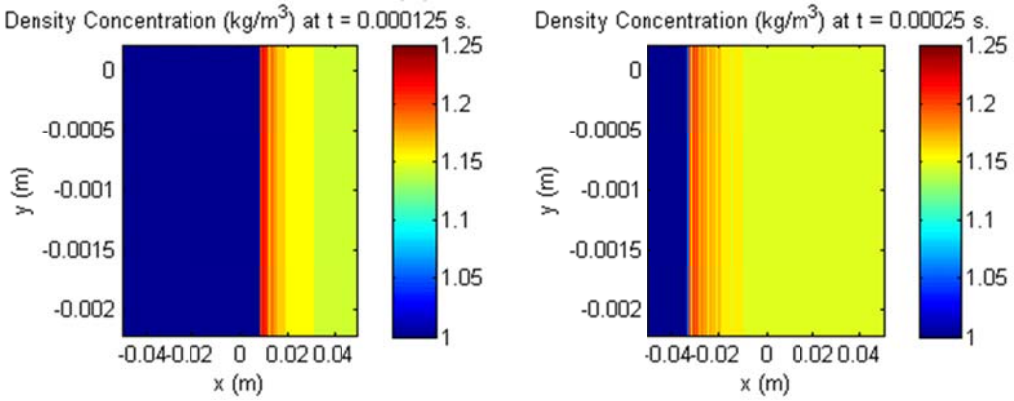

Figure 16-4. Extra fine-size (XF) density solution of the Noh-like problem.

Figure 16 indicates that the transition between the undisturbed and post-shock regions does not change significantly as the mesh is refined. It also shows that the numerical solver preserves the planar symmetry since the discrete solutions do not vary as a function of the Y-axis (shown) and Z-axis (not shown). The profiles of density values are extracted at Y = 0 and analyzed next.

A profile view of the mesh refinement is given in Figure 17, where the four discrete solutions are compared to the exact solution derived in Appendix B. The comparison shows that both location of the discontinuity and post-shock density agree very well with the exact solution. The Gibbs oscillations that originate from the numerical truncation are clearly visible. This phenomenon is similar to the truncation of a Fourier expansion when approximating a discontinuous function. As the mesh is refined, the Gibbs oscillations die out quicker and their amplitudes increase. These observations are consistent with how the Gibbs phenomenon behaves.

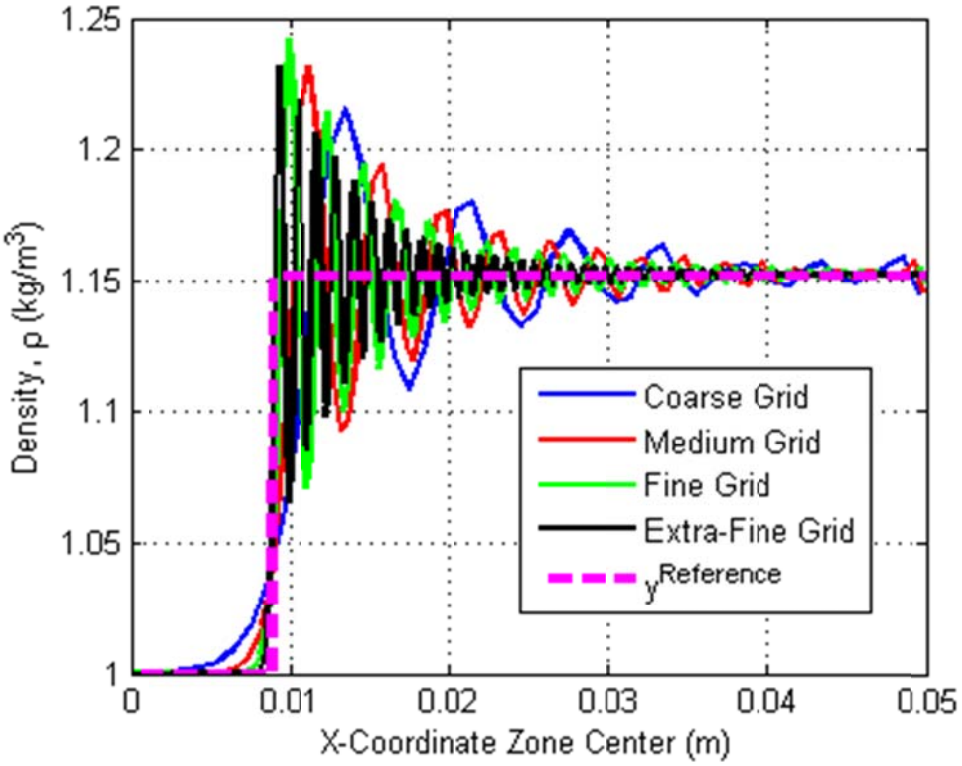

Figure 17. Observation of the Gibbs phenomenon at t = 125 micro-sec.

The oscillations observed in Figure 17 can be explained physically. When a shock discontinuity advances into the fluid, it stops motion of the small element of fluid in its immediate vicinity. The flow then attempts to convert this kinetic energy of the fluid element into internal energy. But the HIGRAD solver does not implement any model of artificial viscosity, meaning that the flow has no mechanism available to dissipate this energy. Since the energy cannot be thermalized by the moving shock, the simulation has no choice but to store it into a "spurious" mode defines as low-amplitude, non-physical fluctuations of the velocity field. These fluctuations, in turn, generate the Gibbs oscillations observed for the density and other state variables. In the absence of models of artificial viscosity, the solutions obtained by the HIGRAD solver behave exactly as expected.

The norms of solution error are listed in Table 9 for the Noh-like problem. These values are the RMS errors $||y^{Exact} - y(\Delta x)||_2$ of density profiles shown in Figure 17. As expected, and despite the presence of Gibbs oscillations, the error decreases as the level of resolution increases.

Table 9. RMS errors of density for the Noh-like problem.

Mesh Used	Mesh Size	Number of Zones, N_X	RMS Errors (x 10^{-3} kg/m^3)
Coarse (C)	Δx = 1 mm	100	15.44
Medium (M)	Δx = 0.5 mm	200	12.61
Fine (F)	Δx = 0.25 mm	400	8.45
Extra Fine (XF)	Δx = 0.125 mm	800	4.96

Table 10 gives the parameters (p; β) of the model of truncation error $||y^{Exact} - y(\Delta x)||_2 = \beta\, \Delta x^p$. The model is best-fitted to the four RMS errors listed in Table 9. β represents the regression pre-factor and p is the observed rate-of-convergence. The analysis of mesh refinement results suggests a sub-optimal convergence because p = 0.55 is observed while a 1st-order accuracy is expected, in theory, for the approximation of a discontinuous solution by the numerical method. The observation of a lower than expected accuracy would normally raise concern about the quality of the implementation or, worst, suggest the presence of a programming mistake. This is, however, not the case here for the following reasons.

Table 10. Parameters of the global model $||y^{Exact} - y(\Delta x)||_2 = \beta\, \Delta x^p$ of solution error.

Rate-of-convergence	Pre-factor Coefficient
p = 0.55	β = 0.321 kg/m^3

First, it is emphasized that the Noh problem is a challenging test for hydrodynamics solvers, whether the conservation laws are formulated in "moving" (Lagrangian) or "stationary" (Eulerian) frames-of-reference. It is common to get rates-of-convergence within ½ ≤ p ≤ ¾ when analyzing this problem. The accuracy estimated in Table 10 matches what is often observed. Second, it is noted that the HIGRAD solver is not constructed to simulate compressible flows that may develop shocks and other types of discontinuities. This is apparent from the occurrence of Gibbs oscillations in Figure 17, a well-understood phenomenon that a shock-capturing method would attempt to suppress with the addition of various forms of artificial viscosity. The presence of these spurious waveforms is undeniably detrimental to the numerical accuracy that we are attempting to estimate in Table 10 using a global model of truncation error.

To further investigate the accuracy of the HIGRAD solver when applied to the Noh problem, the functional data analysis of Reference [19] is used next. This technique is capable of analyzing the self-convergence of entire fields, such as the density profiles of Figure 17, in the sense of the L^2 norm. It also offers the advantage of decomposing the discrete solutions on a truncated basis of empirical "modes." The orthogonal decomposition makes it possible to analyze the self-convergence mode-by-mode, which greatly simplifies the derivations. It also provides a way to filter out undesirable waveforms that may "pollute" the solutions or be the manifestation of unstable contributions, such as the Gibbs oscillations of the Noh problem. For completeness, the main steps of the procedure are briefly described next. It is shown that, when this analysis technique is applied, a nearly 2nd-order accuracy is recovered for most of the density solution. The results suggest that the HIGRAD solver handles well the simulation of a discontinuous flow.

The functional analysis of self-convergence starts by collecting the discrete solutions in a single data matrix denoted by the symbol Y below. The discrete solutions are the profiles of density obtained by running the Noh problem with different levels of mesh resolution. The data matrix Y collects N_D rows (number of sample points) and m columns (number of computer runs) as:

$$\underset{N_D-by-m}{Y} = \begin{bmatrix} y_1(\Delta x_1) & y_1(\Delta x_2) & \cdots & y_1(\Delta x_m) \\ y_2(\Delta x_1) & y_2(\Delta x_2) & \cdots & y_2(\Delta x_m) \\ \vdots & \vdots & \ddots & \vdots \\ y_{N_D}(\Delta x_1) & y_{N_D}(\Delta x_2) & \cdots & y_{N_D}(\Delta x_m) \end{bmatrix}, \tag{26}$$

where $y_s(\Delta x)$ denotes the s^{th} sample ($s = 1 \ldots N_D$) of the solution obtained with mesh size Δx.

One practical, as well as theoretical, issue is that the discrete solutions $y(\Delta x)$ must be vectors of same length. This is not the case since these solutions come from simulations performed with different levels of resolution. This is handled by re-sampling the density profiles of Figure 17 on a common "reference" discretization. The 1-mm grid, with 100 zones defined in the X-direction of the flow, is used for this operation that consists of volume-averaging the discrete solutions on this coarser "reference." Other methods to re-sample the discrete solutions are discussed in the reference.[19] For theoretical reasons not discussed here, the averaging scheme is the technique that should be implemented whenever possible. The re-sampled data are shown in Figure 18.

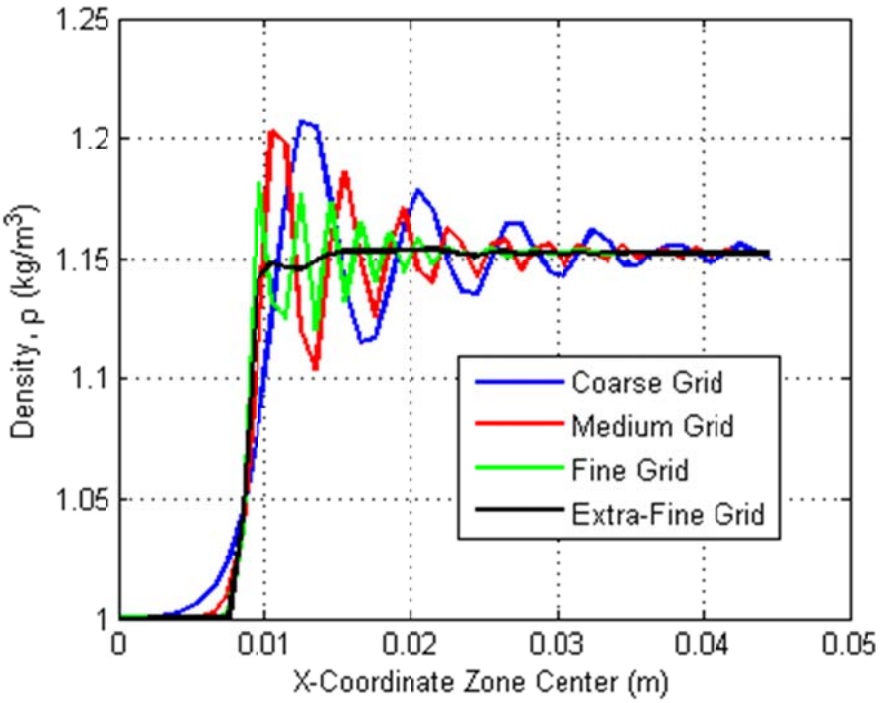

Figure 18. Re-sampled solutions for functional analysis using Δx = 1 mm (100 zones).

The next step is to perform a Singular Value Decomposition (SVD) of the data matrix, which can be written compactly as:

$$\underset{N_D-by-m}{Y} = \underset{N_D-by-m}{U} \cdot \underset{m-by-m}{\Sigma} \cdot \underset{m-by-m}{V}^T. \tag{27}$$

In equation (27), the symbols U and V denote the orthogonal matrices of left and right singular vectors, respectively, and Σ is the diagonal matrix singular values. The triplet of singular values and left and right vectors (σ_k; U_k; V_k) represents the k^{th} mode of the decomposition. This method is analogous to a Principal Component Decomposition (PCD) where the objective is to identify the most significant trends of a dataset and separate them from secondary effects.

Table 11. Singular values σ_k of the principal component decomposition.

Mode	Singular Value, σ_k	Scaled Value, $\sigma_k/\Sigma\sigma_k$	Information Content, σ_k^2
1	20.188 kg/m^3	98.64%	99.992%
2	0.132 kg/m^3	0.65%	99.997%
3	0.110 kg/m^3	0.54%	99.999%
4	0.035 kg/m^3	0.17%	100.000%

The singular values of the decomposition represent the importance of each mode to reconstruct the discrete solutions. They are listed in Table 11 for the analysis of re-sampled densities shown in Figure 18. It is observed that the first mode captures over 99.99% of the information content, which is defined as the summation of squared singular values (last column of Table 11). This indicates that, for all practical purposes, analyzing the projections of discrete solutions in the direction of this dominant first mode U_1 should suffice to assess the ability of the HIGRAD solver to self-converge. Another observation is that modes 2-to-4 may indicate numerical "noise" since their overall contributions are so weak. This is often observed when low-level perturbations are introduced in a simulation, for example, by a contact condition between two surfaces in solid mechanics, an interface between two different materials in fluid dynamics, or the violation of a stability condition of the integration method. The significance of these modes is verified next.

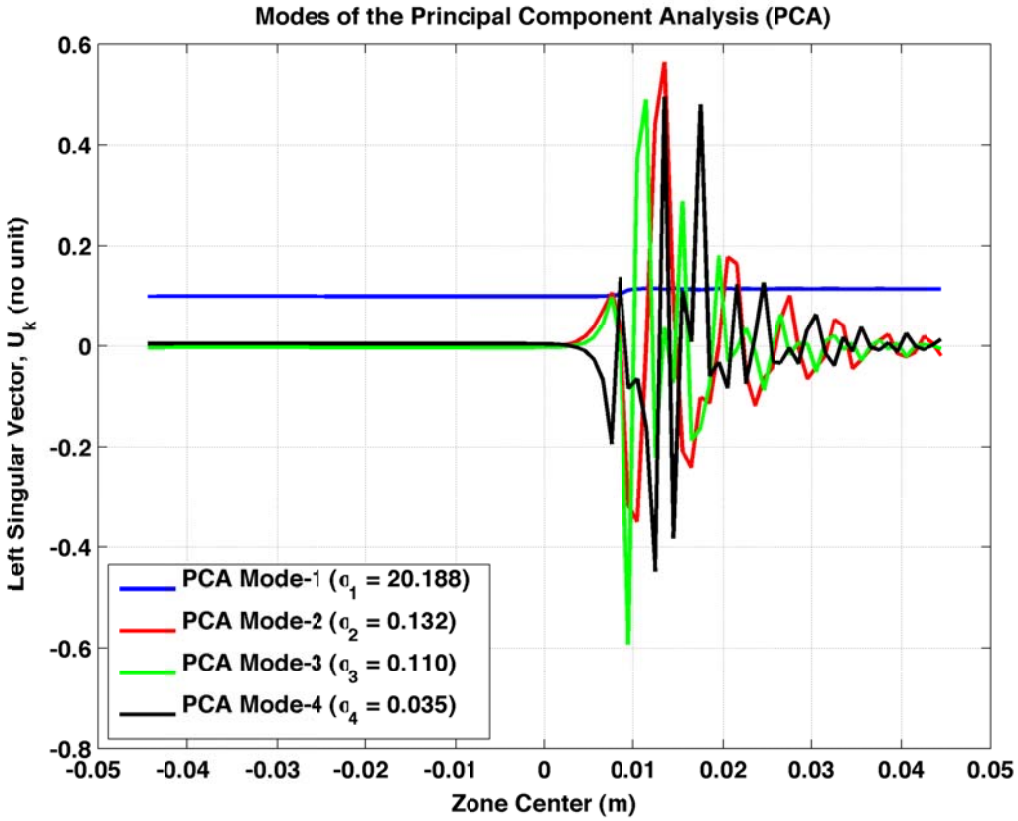

Figure 19. Left singular vector U_k of the principal component decomposition.

Figure 19 depicts the left singular vectors U_k of the decomposition. The vectors are by definition expressed on the same discretization as the data matrix Y, which is sampled on 100 zones. The

vectors are dimensionless and shown without accounting for their respective importance, which is measured by the corresponding singular values σ_k. Figure 19 indicates that, as expected, the first SVD mode captures the undisturbed and post-shock regions of the flow. As the order of the decomposition increases, the subsequent modes attempt to capture finer-resolution details. The second mode matters most at the location of the discontinuity, without attempting to represent the Gibbs oscillations away from the shock front. Modes 3 and 4 represent finer-scale details of the post-shock oscillations. Understanding the nature of these various modes matters because it provide insight into the convergence behavior of the discrete solutions.

The orthogonal vectors (U_1; U_2; U_3; U_4) form a basis of the sub-space defined by the columns of matrix Y. They can be used to express any discrete solution such as:

$$y(\Delta x) = \sum_{k=1...m} \eta_k(\Delta x)\, U_k \ , \tag{28}$$

where the scalar value $\eta_k(\Delta x)$ denotes the general coordinate defined by projecting the solution $y(\Delta x)$ in the direction of the k^{th} left singular vector U_k (with an inner product notation below):

$$\eta_k(\Delta x) = U_k^T\, y(\Delta x). \tag{29}$$

Alternatively, it can be verified the generalized coordinates can be calculated simply as:

$$\eta_k(\Delta x) = \sigma_k\, V_k(\Delta x). \tag{30}$$

Because the generalized coordinate $\eta_k(\Delta x)$ is a function of grid resolution, for each mode of the decomposition, self-convergence can be studied in a manner similar to the application of mesh refinement to any scalar response of the code. This is convenient because the analysis can be performed independently from one mode to the next. Results hence obtained are summarized in Table 12. Four models of truncation error, $|\eta_k^{Reference} - \eta_k(\Delta x)| = \beta_k\, \Delta x^{p}{}_k$, are best-fitted to the generalized coordinates. The main difference with similar analyses performed previously is that the triplet of unknowns ($\eta_k^{Reference}$; β_k ; p_k) now depends on the principal component mode.

Table 12. Solution convergence of generalized coordinates of the decomposition.

Mode	Rate-of-convergence	Regression Pre-factor	Extrapolated Solution
1	$p_1 = 1.55$	$\beta_1 = 238.129$ kg/m^3	$\eta_1^{Reference} = -10.092$ kg/m^3
2	$p_2 = 1.93$	$\beta_2 = 90{,}216.10$ kg/m^3	$\eta_2^{Reference} = 32.496\ 10^{-3}$ kg/m^3
3	$p_3 = -1.27$	$\beta_3 = 2.619\ 10^{-6}$ kg/m^3	$\eta_3^{Reference} = -20.183\ 10^{-3}$ kg/m^3
4	$p_4 = -0.98$	$\beta_4 = 4.955\ 10^{-6}$ kg/m^3	$\eta_4^{Reference} = 0.146\ 10^{-3}$ kg/m^3

The first observation from Table 12 is the better-than-first-order accuracy of the first and second modes of the decomposition. The second mode even indicates a second-order accuracy. The reason why the rate-of-convergence of the dominant mode is not closer to the theoretical value of p = 2 is likely because the undisturbed and post-shock regions are well captured by the four discrete solutions, even with the coarsest level of resolution. Their projections in the direction of mode U_1, therefore, are very similar. The authors have observed this phenomenon in other studies: analyzing nearly identical generalized coordinates $\eta_k(\Delta x)$ deteriorates the convergence below expectation. What is essential is that the first two modes of the decomposition indicate better-than-first-order accuracy, as opposed to Table 10 where the “conventional” analysis of L^2 norm of solution error seems to suggest sub-optimal accuracy.

The second observation from Table 12 is that the third and fourth modes, which capture finer-scale details of post-shock oscillations, are actually unstable. This is clear from their negative rates-of-convergence; it means that the solution error grows as the level of resolution increases in the simulation. Combining the visual illustration of Figure 19 with the analysis of asymptotic behavior leaves no doubt that these components are introduced by the Gibbs oscillations and, therefore, not expected to converge. Ignoring these unstable components is the right thing to do because the theory of asymptotic convergence applies only to stable solutions.

Recalling that the two stable modes account for over 99.997% of the information, the functional data analysis provides strong evidence that the numerical performance of the HIGRAD solver matches expectation for this variant of the Noh problem. The nearly second-order accuracy is observed because the re-sampling algorithm illustrated in Figure 18 “smoothens” the shock over several zones, which mitigates the sharp discontinuity that would otherwise reduce any order of accuracy to first-order. The presence of a programming mistake, that could be suspected based on the results of Table 10, can be ruled out.

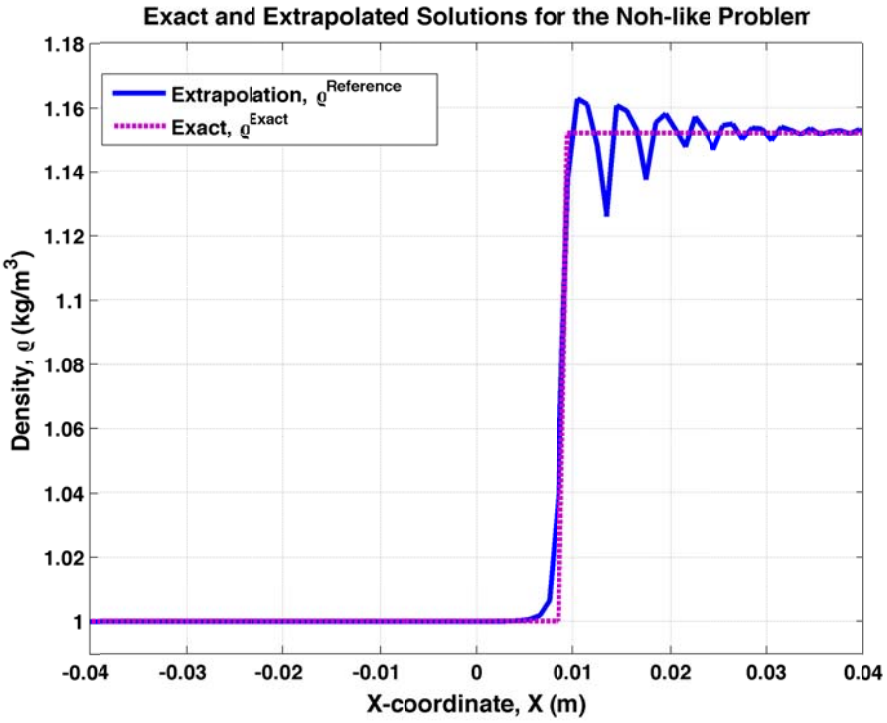

Figure 20. Comparison of exact and extrapolated solutions for the density field.

Finally, the exact and extrapolated solutions are compared in Figure 20. Because the functional data analysis assesses self-convergence without requiring knowledge of the exact solution, the latter provides an independent piece of information that can be used to verify the extrapolation. The extrapolated solution is obtained from equation (28) and using the extrapolated coordinates of Table 12. The reconstruction is limited to the stable modes. Figure 20 pictures an excellent agreement for features of the flow such as the location of the discontinuity and average post-shock condition. The RMS error between these two density solutions is 6.0×10^{-3} kg/m^3. This final step verifies the ability of the HIGRAD code to accurately self-converge the solutions.

8. Conclusion

This report provides documentation of verification activities deployed on the HIGRAD code of computational fluid dynamics used to model atmospheric hydrodynamics and flow conditions. Highly accurate, analytical solutions are derived for four verification test problems that exercise different aspects of the code: (i) quiet start, (ii) passive advection, (iii) passive diffusion, and (iv) a Noh-like problem of compressible flow. The analytical solutions provide "references" useful to, first, verify the lack of obvious programming mistake and, second, quantify the overall accuracy of numerical solutions. In some cases, temporal or spatial refinement studies are carried out to assess the numerical performance of algorithms implemented in HIGRAD by comparing the observed rates-of-convergence to their theoretical counterparts. The quantitative nature of these verification activities is an asset to establish the "pedigree" and, hence, credibility of the code.

The reference solutions for the first two test problems are inferred from basic physics principles. The quiet start problem behaves as expected, without introducing any error in the solution. The passive advection problem indicates nearly second-order accuracy. The observation of steady, monotonic convergence suggests that the solver is properly implemented for this problem. Not matching exactly the theoretical order of accuracy could be due to the treatment of the boundary condition where discretization of the gradient operator may only be first-order accurate. It does not, in our view, "fail" the verification exercise. The most significant implication of a sub-optimal accuracy is the tendency to increase the numerical uncertainty due to truncation effects.

A time convergence study is also performed using the passive advection problem. It provides inconclusive results about the behavior of the time-differencing schemes because the three Runge-Kutta methods tested indicate first-order accuracy. A confounding factor of these studies may be the CFL-based, stability condition that makes it difficult to "isolate" the truncation effects due to temporal discretization from those of spatial discretization. It is recommended to further investigate the time-differencing schemes implemented in HIGRAD to resolve the inconsistency.

The analytical solution for the passive diffusion problem is derived using Fick's second law. The linear diffusion of this problem is modeled by a Laplacian operator that produces "smooth" and well-behaved solutions, leading to the expectation of second-order accuracy. The convergence rates estimated from several spatial refinement studies are consistent and insensitive to settings of the stability condition and time step control. However, they indicate only first-order accuracy. There is strong evidence that the numerical treatment of the boundary condition deteriorates the accuracy because the second-order behavior is recovered when zones located near the edges of the domain are excluded from the analysis. The diffusion problem shows that significant error is generated near the boundary. It is suggested to perform follow-up studies to better quantify this effect with other test problems and diagnose its source in the discretization of the Laplacian operator and/or numerical treatment of the boundary condition.

Lastly, the HIGRAD code is assessed with a problem whose initial conditions closely mimic the well-known Noh problem commonly used to verify the capability of hydrodynamics methods to handle shocks and discontinuities. Due to the inability to exactly impose the boundary condition needed for the Noh test problem, an adjusted analytical solution is derived. The problem is used to assess the ability of the solver to, first, capture the Rankine-Hugoniot "jump" conditions and, second, convert kinetic energy into internal energy. It is noted that, because the HIGRAD code does not implement any model of artificial viscosity, it is not expected to do "well." This test is nevertheless instructive to observe how the code behaves outside of its regime of applicability.

Despite algorithmic limitations, the HIGRAD solver is capable of predicting features of the flow, such as the velocity of the discontinuity, its location, and overall post-shock state, with accuracy. The lack of dissipation yields "spurious" oscillations that are analogous to Gibbs' phenomenon,

which is expected. Analyzing the behavior of these numerical solutions leads to a convergence rate of p = ½, approximately, under the conditions of uniform spatial refinement. Not recovering the first-order accuracy is not, however, a deficiency of the code as briefly summarized below.

A functional data analysis is applied to the solutions of the Noh-like problem. It decomposes the solutions into empirical modes and exploits their orthogonality property to analyze the behavior of truncation error mode-by-mode. The procedure indicates that the convergence of numerical solutions projected on the first two modes, which represents over 99.997% of the information, is nearly second-order accurate. The analysis also suggests that the other modes, while they represent less than 0.003% of the information, are unstable because they attempt to capture the Gibbs oscillations. In conclusion, the functional data analysis indicates that HIGRAD behaves as expected when simulating compressible flows that may develop an evolving discontinuity.

The code verification activities performed herein are an encouraging step towards establishing the “pedigree” and, therefore, credibility of the HIGRAD code. No major flow is diagnosed. This conclusion comes with the important caveat that the verification problems analyzed only provide a partial level of coverage of the overall capabilities, and potential applications, of HIGRAD. Test problems capable of exercising other aspects of the code that are not evaluated herein, such as thermal effects, should be investigated.

In conclusion, it is recommended that the time-differencing schemes implemented be studied to better understand why their accuracy seems to be limited to first-order. Assessing the numerical treatment of the boundary condition, and potential interaction with the discretization of operators such as the gradient and Laplacian, should also be considered for further investigation. For the simulation of flow conditions around wind turbines, future efforts should consider the WindBlade code. In particular, the coupling between the HIGRAD and NLBeam codes should be examined. Finally, it is suggested that these verification activities be systematically captured by the code development team in a regression suite. Once a test problem is developed and analyzed, it is relatively easy to script it to progressively build a regression suite that can be analyzed in a daily or weekly basis. Doing so is essential to provide evidence that the models and algorithms perform according to expectation as new versions are developed and released.

9. Acknowledgements

This work is performed under the auspices of the Laboratory Directed Research and Development project “Intelligent Wind Turbines” at Los Alamos National Laboratory (LANL). The authors are grateful to Curtt Ammerman, project leader, for his support. The support of Rod Linn is also acknowledged and greatly appreciated. LANL is operated by the Los Alamos National Security, LLC for the National Nuclear Security Administration of the U.S. Department of Energy under contract DE-AC52-06NA25396.

10. Bibliographic References

[1] Smith, W.S., Linn, R.R., Sauer, J.A., Canfield, J.M., Winterkamp, J.L., "HIGRAD/FIRETEC Wildfire Model: Description of Basic Theory and Numerics," *Technical Report LA-UR-07-8232*, Los Alamos National Laboratory, Los Alamos, New Mexico, 2007.

[2] Dalton, S., Monahan, L., Stevenson, I, Luscher, D.J., Park, G., Farinholt, K, "Experimental Assessment of NLBeam for Modeling large Deformation Structural Dynamics," *30th SEM International Modal Analysis Conference*, Jacksonville, Florida, January 30-February 2, 2012.

[3] Laird, D.L., Montoya, F.C., Malcolm, D., "Finite Element Modeling of Wind Turbine Blades," *43rd AIAA Aerospace Sciences Meeting and Exhibit*, Reno, Nevada, January 10-13, 2005.

[4] Grinstein, F.F., Margolin, L.G., Rider, W.J., **Implicit Large Eddy Simulation**, *Cambridge University Press Publishers*, New York, New York, 2007.

[5] Reisner, J.M., Mousseau, V.A., Wyszogrodzki, A.A., Knoll, D.A., "An Implicitly Balanced Hurricane Model with Physics-based Preconditioning," *Monthly Weather Review*, Vol. 133, 2005, pp. 1003-1022.

[6] Reisner, J.M., Jeffery, C.A., "A Smooth Cloud Model," *Monthly Weather Review*, Vol. 137, 2009, pp. 1825-1843.

[7] Reisner, J.M., Wynne, S., Margolin, L., Linn, R.R., "Coupled Atmospheric-Fire Modeling Employing the Method of Averages," *Monthly Weather Review*, Vol. 128, 2000, pp. 3686-3691.

[8] LeVeque, R.J., **Numerical Methods for Conservation Laws**, *Lectures in Mathematics, Birkhäuser-Verlag Publishers*, Zürich, Switzerland, Second Edition, 1992.

[9] Lax, P.D, Wendroff, B., "Systems of Conservation Laws," *Communications in Pure and Applied Mathematics*, Vol. 13, 1960, pp. 217-237.

[10] Kuzmin, D., Löhner, R., Turek, S., **Flux-Corrected Transport: Principles, Algorithms, and Applications**, *Springer-Verlag Publishers*, Berlin, Germany, 2005.

[11] Hairer, E., Lubich, C., Roche, M., **The Numerical Solution of Differential-Algebraic Systems by Runge-Kutta Methods**, *Lecture Notes in Mathematics, Springer-Verlag Publishers*, Berlin, Germany, 1989.

[12] Hirt, C.W., "Heuristic Stability Theory for Finite Difference Equations," *Journal of Computational Physics*, Vol. 2, 1968, pp. 339-355.

[13] Warming RF. Hyett BJ. "The Modified Equation Approach to the Stability and Accuracy Analysis of Finite Difference Methods," *Journal of Computational Physics*, Vol. 14, 1974, pp. 159-179.

[14] Lax, P.D., Richtmyer, R.D., "Survey of the Stability of Linear Finite Difference Equations," *Communications in Pure and Applied Mathematics*, Vol. 9, 1956, pp. 267-293.

[15] Smitherman, D.P., Kamm, J.R., Brock, J.S., "Calculation Verification: Point-wise Estimation of Solutions and Their Method-associated Numerical Error," *Technical Report LA-UR-05-8002*, Los Alamos National Laboratory, Los Alamos, New Mexico, 2005.

[16] Hemez, F.M., "Non-linear Error Ansatz Models for Solution Verification in Computational Physics," *Technical Report LA-UR-05-8228*, Los Alamos National Laboratory, Los Alamos, New Mexico, 2005.

[17] Roache, P.J., **Verification and Validation in Computational Science and Engineering**, *Hermosa Publishers*, Albuquerque, New Mexico, 1998.

[18] Hemez, F.M., Hutchens, G.J., Bos, R.J., "Verification of Mesh Adaptation Used in Lagrangian Hydrodynamics Calculations," *Technical Report LA-UR-08-6011*, Los Alamos National Laboratory, Los Alamos, New Mexico, 2008.

[19] Hemez, F.M., "Functional Data Analysis of Solution Convergence," *Technical Report LA-UR-07-5758,* Los Alamos National Laboratory, Los Alamos, New Mexico, 2007.

Appendix A: Derivation of the Analytical Solution for the Passive Scalar Diffusion Test Problem

Physically, diffusion describes the process by which particles spread from regions of higher to lower concentrations. The diffusion phenomenon simulated numerically in the test problem involves defining an essential boundary condition at the top and bottom of the domain, where the coordinates are z = 0 m and z = 64 m, respectively. Side edges of the rectangular domain are defined with a natural boundary condition, that is, a boundary where the time rate of change of the Oxygen concentration is zero.

Table A-1 defines the variables and coefficients involved in the derivation of the exact solution. The table also proposes a set of units. The units can be changed to another set, as long as it remains internally consistent.

Table A-1. Variables and coefficients needed to define the diffusion test problem.

Symbol	Physical Meaning	Physical Units
q	Oxygen concentration	kg/m^3
κ	Diffusivity constant	no unit
(x; y; z)	Spatial coordinates	meter (m)
t	Time	second (s)
$\mathbf{q_\infty}$	Oxygen concentration diffused	kg/m^3
Q	Initial Oxygen concentration	kg/m^3
$\mathbf{L_x, L_y, L_z}$	Length of domain	meter (m)

In Cartesian three dimensions (x; y; z), Fick's second law of diffusivity can be used to predict how the concentration changes with time:

$$q_t - \kappa\left(q_{xx} + q_{yy} + q_{zz}\right) = 0; \quad q \equiv q(x, y, z, t) \tag{A-1}$$

where q is the Oxygen concentration, t is the time, κ is the diffusivity constant, and (x; y; z) are the dimensions of length in each direction. The subscript t indicates the first partial derivative with respect to time and subscripts xx, yy, and zz indicates second partial derivatives taken with respect to the given variables. The initial condition is defined in equation (A-2):

$$\begin{gathered} q_x(0, y, z, t) = q_x(L_x, y, z, t) = 0 \\ q_y(x, 0, z, t) = q_y(x, L_y, z, t) = 0 \\ q_z(x, y, 0, t) = 0,\ q(x, y, L_z, t) = q_\infty \\ q(x, y, z, 0) = Q(x, y, z) \end{gathered} \tag{A-2}$$

Here, the lengths of the domain in the x, y, and z directions are denoted as L_x, L_y, and L_z, respectively. The symbol Q represents the initial Oxygen concentration in the domain, and q_∞ is the Oxygen concentration being diffused into the domain. The initial condition of equation (A-2) specifies an Oxygen concentration diffused in the z direction only, and at the top of the domain.

The problem is reduced to a one-dimensional equation in the z direction. This equation does not depend on the x and y coordinates because of the use of a natural boundary condition:

$$q_t - \kappa\left(q_{zz}\right) = 0; \quad q \equiv q(z, t) \tag{A-3}$$

with the following initial condition:

$$q_z(0,t) = 0 \qquad q(L_z,t) = q_\infty \qquad q(z,0) = Q(z) \tag{A-4}$$

To solve the system of equations (A-3) and (A-4), the following transformation is made to obtain homogenous boundary conditions:

$$q = \varphi + q_\infty; \quad \varphi \equiv \varphi(z,t) \tag{A-5}$$

This substitution yields the following equations, in which all right-hand sides are equal to zero:

$$\begin{gathered} \varphi_t - \kappa(\varphi_{zz}) = 0 \\ \varphi_z(0,t) = 0, \quad \varphi(L_z,t) = 0 \end{gathered} \tag{A-6}$$

Next, the separation of space and time variables (z; t) is invoked to obtain a separable solution:

$$\varphi(z,t) = Z(z)T(t) \tag{A-7}$$

Substituting the decomposition (A-7) into equations (A-5) and (A-6) leads to the four equations:

$$\begin{gathered} \frac{T'}{\kappa T} = \frac{Z''}{Z} = -\tau \\ Z'(0) = 0, \quad Z(L_z) = 0, \quad T'(0) = 0 \end{gathered} \tag{A-8}$$

Variables in equation (A-8) can be further separated such that two distinct ordinary differential equations are formulated, as shown in equation (A-9):

$$\frac{T'}{\kappa T} = -\tau \qquad \frac{Z''}{Z} = -\tau \tag{A-9}$$

where:

$$\tau = \lambda_z^2 \tag{A-10}$$

First, consider the ordinary differential equation for *Z* using the transformed initial condition:

$$\begin{gathered} Z'' + \lambda_z^2 Z = 0 \\ Z'(0) = Z(L_z) = 0 \end{gathered} \tag{A-11}$$

which has the solution:

$$Z = A_l \cos(\lambda z); \quad \lambda = \frac{(2l-1)\pi}{2L_z}; \quad l = 1,2,3,\ldots \tag{A-12}$$

Second, consider the ordinary differential equation for *T* using the transformed initial condition:

$$\begin{gathered} T' + \tau\kappa T = 0 \\ T'(0) = 0 \end{gathered} \tag{A-13}$$

which has the solution:

$$T = E_l \exp(-\kappa\tau_l t) \tag{A-14}$$

Next, expressions (A-7), (A-12), and (A-14) are substituted in equation (A-5) to yield a solution for the Oxygen concentration q expressed as an infinite series expansion:

$$q = q_\infty + \sum_{l=1}^{\infty} A_l \cdot e^{-\kappa\tau_l t} \cdot \cos(\lambda z). \tag{A-15}$$

The orthogonality of Fourier series is used to define and calculate the coefficients A_l:

$$A_l = \frac{2}{L_z}\int_0^{L_z} (Q - q_\infty)\cos(\lambda z)dz \tag{A-16}$$

which leads to:

$$A_l = \frac{4(Q - q_\infty)}{(2l-1)\pi}\sin\left(\frac{(2l-1)\pi}{2}\right) \tag{A-17}$$

Equation (A-17) can be written in a more compact form as:

$$A_l = \frac{4(Q - q_\infty)}{(2l-1)\pi}(-1)^{l-1}. \tag{A-18}$$

The final step is to substitute the coefficient A_l of equation (A-18) in the solution (A-15). The exact solution can be used to calculate the Oxygen concentration at any time t > 0:

$$q = q_\infty + \sum_{l=1}^{\infty} \frac{4(Q - q_\infty)}{(2l-1)\pi}(-1)^{l-1} \cdot e^{-\kappa\tau_l t} \cdot \cos\left(\frac{(2l-1)\pi z}{2L_z}\right). \tag{A-19}$$

Appendix B: Derivation of the Analytical Solution for the Noh-like Test Problem Based on Rankine-Hugoniot "Jump" Conditions in 1D, Cartesian Geometry

For reasons explained below, the test problem is a variant of the well-known Noh problem. The conventional Noh problem is illustrated in Figure B-1 for a Lagrangian hydrodynamics mesh. The velocity field is initialized as uniform and constant across the computational domain. This motion compresses the fluid, progressively converting kinetic energy into internal energy. Because the right boundary is fixed and non-diffusive, zones on the right side start to compress immediately. Their density increases by a related amount, which expresses the conservation of mass and momentum. It generates a discontinuity in density and pressure that propagates from right to left. Figure B-1 shows the initial computational mesh at time zero (left) and the deformed mesh at 10^{-3} sec. (right). Details about this, and related, problems are found in Reference 18.

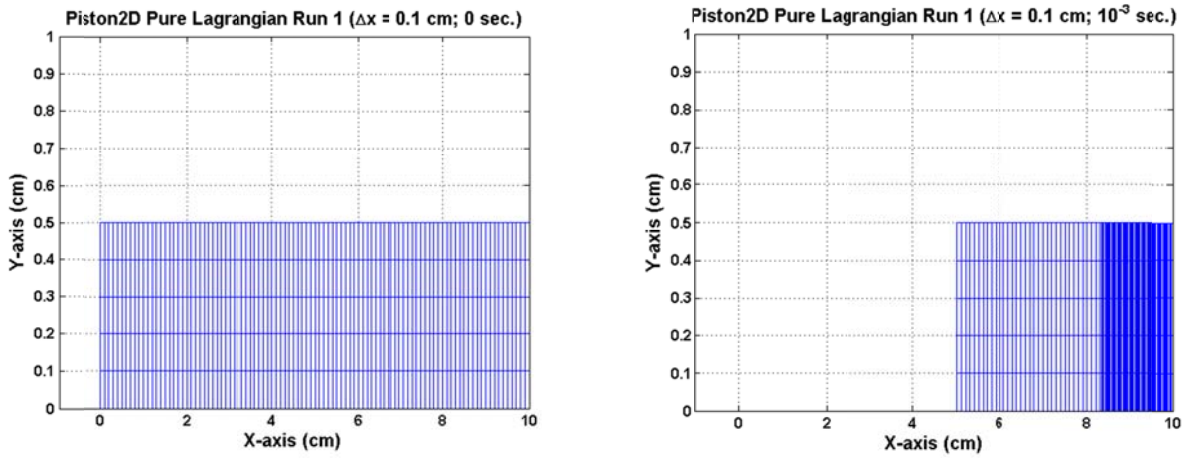

Figure B-1. Noh test problem on a Lagrangian mesh. (Reference: LA-UR-08-6011.)

The Noh problem tests two basic properties of a hydrodynamics code. Because motion comes to a halt behind the discontinuity, the problem assesses, first, the ability of the code to convert kinetic energy into internal (or potential) energy. Second, it assesses the ability of the code to capture properties of the discontinuity, also known as the Rankine-Hugoniot "jump" conditions.

The Computational Fluid Dynamics (CFD) solver investigated in this study relies on a Eulerian frame-of-reference that does not move with the flow. In addition, the boundary implemented is such that fluid flows through the edge of the domain. Because the problem is defined without a perfectly fixed and non-diffusive boundary condition on the right side of the rectangular domain, the conventional solution of the Noh problem cannot be used for code verification. This is the reason why a variant of the test problem is derived for comparison with the numerical solutions computed by the HIGRAD code.

The Noh-like test problem proposed below focuses on the ability of the CFD solver to satisfy the Rankine-Hugoniot "jump" conditions. This choice is made because the HIGRAD code is not written to handle shocks. It does not, for example, have any mechanism to add artificial viscosity to the numerical solution if a shock is found. The Rankine-Hugoniot "jump" conditions, however, apply whether the discontinuity is strong or weak, such as a sharp shock or smoother transition between two hydrodynamic states spread over several zones of the computational domain. By focusing on flow properties in the vicinity of the discontinuity, an analytical solution is derived that remains valid irrespective of the implementation of the initial and boundary conditions.

B-1 The Rankine-Hugoniot "Jump" Conditions in 1D, Cartesian Geometry

The exact solution derived next is based solely on the "jump" conditions, without consideration for the initial and boundary conditions. It implies that, depending on how these conditions are implemented, the exact solution may not be valid near the edges of the computational domain. Likewise the solution presented next does not apply when the shock reaches, and bounces off, the opposite boundary. The derivation of the exact solution is presented next.

The partial differential equations solved by the CFD solver express conservation laws that can be written in a generic sense as:

$$\frac{\partial q(x;t)}{\partial t} + \nabla F(q) = S(x;t) \tag{B-1}$$

where the symbol *q* denotes a single, or multiple, state variables. The functions *F(•)* and *S(•)* are the flux and source terms, respectively. Equation (B-1) expresses that the rate-of-change in time of quantity *q* is in equilibrium with the spatial gradient of the flux, modulo any amount that the source term adds or takes away. We seek a simplified test problem in one-dimensional (1D), Cartesian geometry. Equation (B-1) can, accordingly, be reduced to:

$$\frac{\partial q}{\partial t} + \frac{\partial F(q)}{\partial x} = 0 \tag{B-2}$$

where the source term is defined as *S(x;t) = 0* for simplicity. Equation (B-2) is further specialized to fluid dynamics next.

In the remainder, we consider an ideal gas that operates under adiabatic conditions. The fluid is assumed inviscid, meaning that the effects of viscosity are negligible. In 1D Cartesian geometry, this fluid can be described by a system of conservation laws known as the Euler equations:

$$\begin{cases} \dfrac{\partial \rho}{\partial t} + \dfrac{\partial (\rho \cdot U)}{\partial x} = 0 \\ \dfrac{\partial (\rho \cdot U)}{\partial t} + \dfrac{\partial (\rho \cdot U^2 + P)}{\partial x} = 0 \\ \dfrac{\partial (\rho \cdot E)}{\partial t} + \dfrac{\partial U \cdot (\rho \cdot E + P)}{\partial x} = 0 \end{cases} \tag{B-3}$$

where the symbols *ρ*, *P*, *U*, and *E* denote the fluid density, pressure, velocity, and specific total energy, respectively. From top to bottom, equation (B-3) expresses the conservation of mass (or continuity equation), momentum, and total energy. Total and internal energies are related by:

$$E = \varepsilon + \frac{1}{2} U^2 \tag{B-4}$$

Combining equations (B-3) and (B-4) indicates that the behavior of the fluid is described by four independent unknowns (*ρ; P; U; ε*) while there are only three conservation laws. Closure of this system of partial differential equations is provided by the equation-of-state (EOS) that relates the density and internal energy states to the pressure state or, conversely, density and pressure to internal energy. In this work, a polytropic EOS described by the following law is assumed:

$$P = (\gamma - 1) \cdot \rho \cdot \varepsilon \tag{B-5}$$

where the symbol γ denotes the adiabatic index (or adiabatic exponent), defined as the ratio of specific heat capacities, or $\gamma = C_P/C_V$. The index γ is a material property of the fluid. The state variables and other coefficients defined in equations (B-3) to (B-5) are summarized in Table B-2, together a consistent set of physical units.

Table B-2. Variables and coefficients needed to define the Noh-like test problem.

Symbol	Physical Meaning	Physical Units
ρ	Mass density	kg/m^3
P	Pressure	Pa ($Pa = N/m^2 = kg\ m^{-1}\ s^{-2}$)
U	Flow/fluid velocity	m/s
ε	Specific internal energy	J/kg ($J = Pa\ m^3 = kg\ m^2\ s^{-2}$)
E	Specific total energy	J/kg
x	Spatial coordinate	meter (m)
t	Time	second (s)
γ	Adiabatic exponent	no unit
C_P, C_V	Specific heat capacities	$J\ kg^{-1}\ {}^oK^{-1}$

Figure B-2 defines the condition of the fluid assumed to formulate the Noh-like test problem. An infinitely strong discontinuity, identified by the subscript $(\bullet)_D$, separates two different states. By analogy with Figure B-1, the discontinuity moves from right to left at the velocity V_D. Fluid located to the left (or "in front") of the discontinuity is in the undisturbed region that has not yet "seen" the discontinuity. Fluid located to the right (or "behind") of the discontinuity is in the post-shock region where motion stops due to the conversion of kinetic energy to internal energy. The undisturbed and post-shock regions are denoted by the subscripts $(\bullet)_1$ and $(\bullet)_2$, respectively.

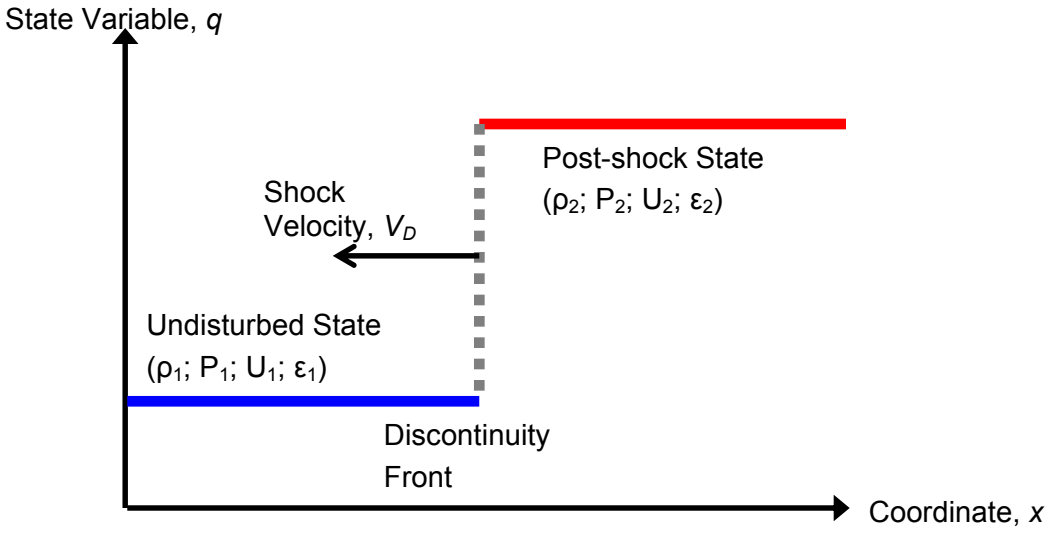

Figure B-2. Undisturbed and post-shock states separated by a discontinuity.

Table B-3 lists the variables defined in Figure B-2 for the undisturbed and post-shock states. The "downstream" state $(\rho_1;\ P_1;\ U_1;\ \varepsilon_1)$ is known because, by definition, the undisturbed region

corresponds to the initial condition controlled by the analyst. The "upstream" state $(\rho_2; P_2; U_2; \varepsilon_2)$ is unknown, together with the velocity V_D at which the discontinuity travels. If a closure equation, such as the polytropic EOS of equation (B-5), is used to calculate the specific internal energy as a function of $(\rho; P)$, then the unknowns reduce to the four variables $(\rho_2; P_2; U_2; V_D)$.

Table B-3. Variables and coefficients needed to define the Noh-like test problem.

Symbol	Physical Meaning	Status
ρ_1	Undisturbed density	Known
P_1	Undisturbed pressure	Known
U_1	Undisturbed flow velocity	Known (and positive)
ε_1	Undisturbed specific internal energy	Known from the pair $(\rho_1; P_1)$
ρ_2	Post-shock density	Unknown
P_2	Post-shock pressure	Unknown
U_2	Post-shock flow velocity	Unknown
ε_2	Post-shock specific internal energy	Known from the pair $(\rho_2; P_2)$
V_D	Velocity of the discontinuity	Unknown (and negative)
γ	Adiabatic exponent	Known

It is further assumed below that the post-shock velocity is known, and equal to $U_2 = 0$, because the variant of the Noh problem studied here is formulated such that kinetic energy (or motion) is converted to internal energy (or heat) through an entropy-producing discontinuity. It reduces the number of unknowns of the problem to the triplet $(\rho_2; P_2; V_D)$.

To derive the "jump" conditions, the conservation laws (B-3) are integrated over a small volume of fluid that extends from one side of the discontinuity to the other side. Because the geometry is 1D Cartesian, the element of fluid is simply an interval denoted by $\Omega = [x_1; x_2]$. Applied to the generic form (B-2) of a conservation law, this integration gives:

$$\int_{x_1}^{x_2} \frac{\partial q}{\partial t} \cdot dx + \int_{x_1}^{x_2} \frac{\partial F(q)}{\partial x} \cdot dx = 0 \tag{B-6}$$

If the first integral converges, then it can be switched with the time derivative to provide:

$$\frac{\partial}{\partial t} \int_{x_1}^{x_2} q \cdot dx + \int_{x_1}^{x_2} \frac{\partial F(q)}{\partial x} \cdot dx = 0 \tag{B-7}$$

The first integral must be separated in two halves, from coordinate $x = x_1$ to the left-side of the discontinuity, then, from the right-side of the discontinuity to coordinate $x = x_2$:

$$\frac{\partial}{\partial t}\left(\int_{x_1}^{x_D^{(-)}} q \cdot dx + \int_{x_D^{(+)}}^{x_2} q \cdot dx \right) + \int_{x_1}^{x_2} \frac{\partial F(q)}{\partial x} \cdot dx = 0 \tag{B-8}$$

Switching, once more, the integral and time derivative operators introduces the derivative of the location x_D of the discontinuity:

$$\left(\int_{x_1}^{x_D^{(-)}} \frac{\partial q}{\partial t} \cdot dx + \int_{x_D^{(+)}}^{x_2} \frac{\partial q}{\partial t} \cdot dx \right) + \left(q_1 - q_2\right) \cdot \frac{\partial x_D}{\partial t} + \int_{x_1}^{x_2} \frac{\partial F(q)}{\partial x} \cdot dx = 0 \tag{B-9}$$

where q_1 and q_2 denote the constant-valued states in the immediate vicinity of the discontinuity:

$$q_1 = \lim_{\substack{x \to x_D^{(-)} \\ x_D^{(-)} \geq x \geq x_1}} q(x) \qquad q_2 = \lim_{\substack{x \to x_D^{(+)} \\ x_2 \geq x \geq x_D^{(+)}}} q(x) \tag{B-10}$$

The final step is to let the element of fluid tend to the discontinuity by taking the limits $x_1 \rightarrow x_D^{(-)}$ on the left-side and $x_2 \rightarrow x_D^{(+)}$ on the right-side. Because the state variables are continuous on both sides of the discontinuity, the first parenthesis of equation (B-9) cancels out:

$$\lim_{\substack{x_1 \to x_D^{(-)} \\ x_D^{(-)} \geq x \geq x_1}} \int_{x_1}^{x_D^{(-)}} \frac{\partial q}{\partial t} \cdot dx = 0 \qquad \lim_{\substack{x_2 \to x_D^{(+)} \\ x_D^{(+)} \geq x \geq x_2}} \int_{x_D^{(+)}}^{x_2} \frac{\partial q}{\partial t} \cdot dx = 0 \tag{B-11}$$

The second and third terms of equation (A-9) reduce to:

$$\left(q_1 - q_2\right) \cdot \frac{\partial x_D}{\partial t} + \left[F(q)\right]_{x=x_1}^{x=x_2} = 0 \tag{B-12}$$

or:

$$\left(q_1 - q_2\right) \cdot \frac{\partial x_D}{\partial t} + \left(F(q_2) - F(q_1)\right) = 0 \tag{B-13}$$

Because the velocity at which the discontinuity travels is defined as $V_D = dx_D/dt$, equation (B-13) leads to the "jump" condition across the discontinuity:

$$V_D = \frac{F(q_2) - F(q_1)}{q_2 - q_1} \tag{B-14}$$

Equation (B-14) can be applied to the system of Euler equations (B-3), which gives:

$$\begin{cases} V_D \cdot \left(\rho_2 - \rho_1\right) = \left(\rho_2 U_2 - \rho_1 U_1\right) \\ V_D \cdot \left(\rho_2 U_2 - \rho_1 U_1\right) = \left(\rho_2 U_2^2 + P_2\right) - \left(\rho_1 U_1^2 + P_1\right) \\ V_D \cdot \left(\rho_2 E_2 - \rho_1 E_1\right) = U_2\left(\rho_2 E_2 + P_2\right) - U_1\left(\rho_1 E_1 + P_1\right) \end{cases} \tag{B-15}$$

The three "jump" conditions expressed in equation (B-15) are manipulated next to get a closed-form solution defined in the immediate vicinity of the discontinuity and away from the boundary.

B-2 The Closed-form Solution of the Noh-like Test Problem

The procedure proposed to derive a closed-form, analytical solution solves the three equations (B-15) simultaneously. By definition of the test problem, the state variables $(\rho_1; P_1; U_1; \varepsilon_1)$ of the undisturbed region are equal to the initial conditions. They are assumed known since the initial conditions are defined and controlled by the user. The system of equations (B-15) is left with the four unknown state variables $(\rho_2; P_2; U_2; \varepsilon_2)$ of the post-shock region, together with the unknown velocity V_D of the discontinuity.

Because we are only seeking to verify the hydro-code with the ideal gas law of equation (B-5), the specific internal energy ε_2 can be solved for once the density ρ_2 and pressure P_2 have been obtained. It reduces the unknowns of equations (B-15) to the quadruplet $(\rho_2; P_2; U_2; V_D)$. Finally, it is assumed that the post-shock velocity is known. This is because the variant of the Noh test problem sought here verifies the ability of the code to convert kinetic energy into internal energy. If this process is represented with accuracy by the numerical solution, then motion should stop behind the discontinuity, which implies that $U_2 = 0$.

The post-shock velocity U_2 is assumed to be equal to zero, a condition that is verified easily by examining the velocity field of the numerical solution computed by the code. It is emphasized that this assumption should be verified independently from the closed-form solution presented. Jointly assuming that the EOS is provided a closed-form equation and the post-shock velocity is equal to zero reduces the unknowns to the triplet $(\rho_2; P_2; V_D)$. It means that (B-15) is a system of three equations with three unknowns. The remainder of this section derives the exact solution.

Substituting the condition $U_2 = 0$ in system (B-15) simplifies the equations:

$$\begin{cases} V_D \cdot (\rho_2 - \rho_1) = -\rho_1 U_1 \\ -\rho_1 U_1 V_D = (P_2 - P_1) - \rho_1 U_1^2 \\ V_D \cdot (\rho_2 E_2 - \rho_1 E_1) = -U_1 (\rho_1 E_1 + P_1) \end{cases} \tag{B-16}$$

Note that the second of equations (B-16) can alternatively be written as:

$$V_D^2 \cdot (\rho_2 - \rho_1) = (P_2 - P_1) - \rho_1 U_1^2 \tag{B-17}$$

by substituting the factor $(-\rho_1 U_1)$ in the left-hand side with the first of equations (B-16). To derive the exact solution, the strategy is to express the unknowns V_D and P_2 as a function of the post-shock density ρ_2. The "jump" condition of the total energy conservation law is then used to solve for the last remaining unknown ρ_2.

We start with the "jump" condition of the continuity equation that can be written as:

$$V_D = -\frac{\rho_1 U_1}{\rho_2 - \rho_1} \tag{B-18}$$

Likewise the "jump" condition of the momentum conservation law can be written by re-arranging the terms of equation (B-17) where the unknown V_D is substituted using equation (B-18):

$$P_2 = P_1 + \frac{\rho_1 \rho_2}{\rho_2 - \rho_1} U_1^2 \tag{B-19}$$

The third of equations (B-16) is the "jump" condition of the total energy conservation law, used next to obtain a closed-form solution for the post-shock density ρ_2:

$$V_D \cdot (\rho_2 E_2 - \rho_1 E_1) = -U_1 (\rho_1 E_1 + P_1) \tag{B-20}$$

The expressions of V_D and P_2 in equations (B-18) and (B-19), respectively, are substituted in equation (B-20) to obtain a single equation that depends only on the unknown density ρ_2:

$$-\frac{\rho_1 U_1}{\rho_2 - \rho_1} \cdot \left(\rho_2 \left(\varepsilon_2 + \underbrace{\frac{U_2^2}{2}}_{\text{Zero}} \right) - \rho_1 E_1 \right) = -U_1 (\rho_1 E_1 + P_1) \tag{B-21}$$

Simplifying the expression leads to:

$$\rho_1 \rho_2 \varepsilon_2 = \rho_1 \rho_2 E_1 + (\rho_2 - \rho_1) P_1 \tag{B-22}$$

Next, the ideal gas EOS is used to replace the specific internal energy:

$$\rho_2 \varepsilon_2 = (\gamma - 1) P_2 \tag{B-23}$$

which, after substitution in equation (B-22), gives:

$$(\gamma-1)\rho_1 P_2 = \rho_1\rho_2 E_1 + (\rho_2-\rho_1)P_1 \tag{B-24}$$

Equation (B-19) is, once more, used to eliminate the post-shock pressure P_2. It gives:

$$(\gamma-1)\rho_1(\rho_2-\rho_1)P_1 + (\gamma-1)\rho_1^2\rho_2 U_1^2 = (\rho_2-\rho_1)\rho_1\rho_2 E_1 + (\rho_2-\rho_1)^2 P_1 \tag{B-25}$$

Finally, the terms of equation (B-25) are rearranged to formulate a quadratic equation in ρ_2:

$$\rho_2^2 \underbrace{[\rho_1 E_1 - P_1]}_{A} - \rho_2 \underbrace{\left[(\gamma+1)\rho_1 P_1 + (\gamma-1)\rho_1^2 U_1^2 + \rho_1^2 E_1\right]}_{B} + \underbrace{\gamma\rho_1^2 P_1}_{C} = 0 \tag{B-26}$$

or:

$$A\rho_2^2 - B\rho_2 + C = 0 \tag{B-27}$$

The two solutions of the quadratic equation are given by:

$$\rho_2 = \frac{B \pm \sqrt{B^2 - 4AC}}{2A} \tag{B-28}$$

given the triplet of coefficients *(A; B; C)* that can be simplified as:

$$\begin{cases} A = (\gamma-2)P_1 + \rho_1\dfrac{U_1^2}{2} \\ B = 2\gamma\rho_1 P_1 + \left(\gamma - \dfrac{1}{2}\right)\rho_1^2 U_1^2 \\ C = \gamma\rho_1^2 P_1 \end{cases} \tag{B-29}$$

A simple test can be implemented to select the solution (B-28) that is positive and greater than the undisturbed density, that is, $\rho_2 \geq \rho_1$. After having obtained the post-shock density, equation (B-18) is used to calculate the velocity V_D of the discontinuity. Likewise equation (B-19) is used to calculate the post-shock pressure P_2. Finally, the EOS (B-23) gives the internal energy.

B-3 Discussion of the Noh-like Test Problem

The procedure to calculate the exact solution is to evaluate, first, the triplet *(A; B; C)* of equation (B-29) knowing the initial condition $(\rho_1; P_1; U_1)$ and adiabatic exponent γ. Second, the quadratic equation is solved by evaluating expression (B-28). The admissible solution, that is, $\rho_2 \geq \rho_1$, is selected. Because of the various assumption made, this closed-form solution is valid only for a "well-developed" discontinuity that is located away from the edges of the computational domain where boundary effects may introduce unwanted perturbations. The closed-form solution is valid only for a polytropic EOS.

The Noh-like test problem is simulated numerically by defining a rectangular domain initialized with a constant velocity everywhere. The initial condition of the fluid $(\rho_1; P_1; \varepsilon_1)$ is also constant throughout the domain. The boundary towards which the velocity vector points is fixed (motion is not allowed) and non-diffusive (no fluid is transferred through the boundary). The opposite-side boundary is fixed in the case of a Eulerian frame-of-reference. For a Lagrangian frame-of-

reference that moves with the flow, the opposite-side boundary moves at the prescribed velocity $U_1 = constant$. The EOS must be a polytropic gas law defined by the adiabatic exponent γ.

A numerical application of equations (B-18), (B-19), (B-28) and (B-29) is presented in Table B-4. It shows that the discontinuity increases the entropy in the system by $\Delta\varepsilon = 36.35$ Joules-per-kg.

Table B-4. Numerical application of the Noh-like test problem.

State	Value	Total Specific Energy
Undisturbed	Density, $\rho_1 = 1$ kg/m^3 Pressure, $P_1 = 1$ Pa Flow velocity, $U_1 = 10$ m/s	$\rho_1(\varepsilon_1 + \frac{1}{2}U_1^2) = 50.40$ J/kg
Post-shock	Density, $\rho_2 = 1.790$ kg/m^3 Pressure, $P_2 = 227.545$ Pa Flow velocity, $U_2 = 0$ m/s	$\rho_2(\varepsilon_2 + \frac{1}{2}U_2^2) = 86.75$ J/kg
Discontinuity	Velocity, $V_D = -12.654$ m/s	Entropy produced: $\Delta\varepsilon = 36.35$ J/kg

(Legend: Numerical application with the adiabatic exponent $\gamma = 1.4$.)

Because of the assumptions made in the derivations, the exact solution can be used only in the vicinity of the discontinuity. It implies that the comparison with a numerical solution computed by the code is justified only if the solution is defined "away" from the edges of the computational domain where the boundary condition, depending on how it is implemented, could violate some of the assumptions made. Finally, it is noticed that the coefficients (B-29) of the exact solution depend on a single free parameter, γ, other than the initial condition. It provides the analyst with the possibility of optimizing this free parameter to "cancel-out" any effect of a boundary condition that may not be exactly implemented as fixed and non-diffusive.

(This report contains 54 pages total.)