LA-UR-17-28532
Approved for public release; distribution is unlimited.

Title: Code Verification Capabilities and Assessments in Support of ASC V&V Level 2 Milestone #6035

Author(s): Doebling, Scott William; Budzien, Joanne Louise; Ferguson, Jim Michael; Harwell, Megan Louise; Hickmann, Kyle Scott; Israel, Daniel M.; Magrogan, William Richard III; Singleton, Robert Jr.; Srinivasan, Gowri; Walter, John William Jr; Woods, Charles Nathan

Intended for: Report

Issued: 2017-09-26 (rev.1)

Code Verification Capabilities and Assessments in Support of ASC V&V Level 2 Milestone #6035

Scott Doebling, Joanne Budzien, Jim Ferguson,
Megan Harwell, Kyle Hickmann, Daniel Israel, William Magrogan,
Robert Singleton, Gowri Srinivasan, John Walter, and Nathan Woods

Los Alamos National Laboratory
Los Alamos, New Mexico 87545, USA

Abstract

This document provides a summary of the code verification activities supporting the FY17 Level 2 V&V milestone entitled "Deliver a Capability for V&V Assessments of Code Implementations of Physics Models and Numerical Algorithms in Support of Future Predictive Capability Framework Pegposts." The physics validation activities supporting this milestone are documented separately. The objectives of this portion of the milestone are: 1) Develop software tools to support code verification analysis; 2) Document standard definitions of code verification test problems; and 3) Perform code verification assessments (focusing on error behavior of algorithms). This report and a set of additional standalone documents serve as the compilation of results demonstrating accomplishment of these objectives.

CONTENTS

I Executive Summary 4
I-A Milestone Completion Criteria and Supporting documentation . . . 4

II Code Verification Theory and Methods 5
II-A Introduction to Code Verification . . . 5
II-B Error Taxonomy . . . 5
II-C Discretization Errors in 2nd-order, shock-capturing codes . . . 5
II-D Use of Exact Solution Problems . . . 6
II-E Use of Error Norms and Convergence Plots . . . 6
II-F The Verification Test Suite . . . 7

III Physics Verification & Validation Infrastructure 8
III-A Introduction: Overview of the Software Solution Provided . . . 8
III-B Lightweight, Simulation, Process, and Data Management (LSPDM) . . . 8
III-C ExactPack . . . 8
III-D EOSlib . . . 9
III-E The Verification Test Suite . . . 10
III-F Software Engineering . . . 10
III-F1 Version Control . . . 10
III-F2 Bug Tracking . . . 10
III-F3 Documentation . . . 10
III-F4 Testing . . . 11
III-F5 Continuous Integration . . . 11

IV Verification of Hydrodynamics 12
IV-A Description of the Problems . . . 12
IV-A1 The Continuum Equations of Motion . . . 12
IV-A2 Noh Problem . . . 12
IV-A3 Sedov . . . 13
IV-A4 Riemann . . . 14
IV-B FLAG Hydrodynamics . . . 18
IV-B1 Summary of Observations . . . 18
IV-C FLAG Sedov Problem . . . 18
IV-C1 Summary of Observations . . . 18
IV-C2 Sedov in FLAG, 1D Spherical, Lagrangian . . . 18

IV-C3 Sedov in FLAG, RZ Cylindrical, Lagrangian . . . 20
IV-C4 Sedov in FLAG, 3D Cartesian, Lagrangian . . . 20
IV-C5 Sedov in FLAG, RZ Cylindrical, ALE . . . 22
IV-D FLAG Riemann Shock Tube Problems . . . 30
IV-D1 Summary of Observations . . . 30
IV-D2 Sod Shock Tube Problem (Riemann Test 1) in FLAG, 1D Cartesian . . . 30
IV-D3 Einfeldt (or 1-2-3) Problem (Riemann Test 2) in FLAG, 1D Cartesian . . . 30
IV-D4 Stationary Contact Problem (Riemann Test 3) in FLAG, 1D Cartesian . . . 30
IV-D5 Slow Shock Problem (Riemann Test 4) in FLAG, 1D Cartesian . . . 30
IV-D6 Shock-Contact-Shock Problem (Riemann Test 5) in FLAG, 1D Cartesian . . . 30
IV-D7 LeBlanc Problem (Riemann Test 6) in FLAG, 1D Cartesian . . . 30
IV-E FLAG Noh Problem . . . 37
IV-E1 Summary of Observations . . . 37
IV-E2 Noh in FLAG, RZ Cylindrical, Lagrangian . . . 37
IV-E3 Noh in FLAG, 3D Cylindrical, Lagrangian . . . 38
IV-F xRAGE Hydrodynamics . . . 39
IV-G xRAGE Sedov Problem . . . 39
IV-G1 Summary of Observations . . . 39
IV-G2 Sedov in xRAGE, 1D Spherical . . . 39
IV-G3 Sedov in xRAGE, 2D-RZ . . . 39
IV-H xRAGE Noh Problem . . . 48
IV-H1 Summary of Observations . . . 48
IV-H2 Noh in xRAGE, 1D Spherical . . . 48
IV-H3 Noh in xRAGE, 2D-RZ . . . 48
IV-I xRAGE Riemann Shock Tube Problems . . . 54
IV-I1 Sod Shock Tube (Riemann Test 1) . . . 54
IV-I2 Einfeldt Problem (Riemann Test 2) in xRAGE, 1D Cartesian . . . 54
IV-I3 Stationary Contact Problem (Riemann Test 3) in xRAGE, 1D Cartesian . . . 54
IV-I4 Slow Shock Problem (Riemann Test 4) in xRAGE, 1D Cartesian . . . 54
IV-I5 Shock-Contact-Shock Problem (Riemann Test 5) in xRAGE, 1D Cartesian . . . 54
IV-I6 LeBlanc Problem (Riemann Test 6) in xRAGE, 1D Cartesian . . . 54
IV-J xRAGE Noh2 Problem . . . 61
IV-J1 Summary of Observations . . . 61

V Verification of Radiative Shocks and an Ion-electron Shock 62
V-A Introduction to radiation hydrodynamics . . . 62
V-B The equilibrium-diffusion radiation model . . . 63
V-C The nonequilibrium-diffusion radiation model . . . 64
V-D The asymptotic-preserving radiative-shock problem . . . 66
V-E The grey S_n-transport radiation model . . . 67
V-F The ion-electron shock problem . . . 68

VI Verification of Material Models 70
VI-A Summary . . . 70
VI-B The Blake Problem . . . 70
VI-B1 Overview of the Blake Problem . . . 70
VI-B2 Blake Results and Analysis . . . 70
VI-C The Hunter Problem . . . 78
VI-C1 Overview of the Derivation of the Hunter Problem . . . 78
VI-C2 Review of Previous Work . . . 79
VI-C3 Hunter Results and Analysis . . . 82

VII Verification of Heat Flow 87
VII-1 Summary of Observations . . . 87
VII-2 The Numerical Issues . . . 87
VII-3 The Planar Sandwich Test Problem . . . 88
VII-4 Convergence Studies of the Planar Sandwich . . . 89
VII-5 The Arithmetic and Harmonic Averages . . . 90
VII-6 Thin Mesh and Static Condensation . . . 91

References 95

I. Executive Summary

A. Milestone Completion Criteria and Supporting documentation

Below are the official milestone completion criteria with a list of the project documents that serve as evidence that each objective has been achieved.

- A repository and software infrastructure to conduct physics verification and validation assessments will be in place. Summary of supporting documents:
  - Physics Verification Infrastructure [Section III of this report]
  - ExactPack Documentation [1]
  - LSPDM Documentation [2]
  - Presentation slides from Level 2 milestone review [3]
- Suites of verification problems and suites of simulation set-ups for the purpose of code physics validation will be in place. Summary of supporting documents:
  - Standardized Definitions for Code Verification Test Problems [4]
  - Three journal articles describing code verification problems for radiation hydrodynamics [5]–[7]
  - One journal article describing a code verification problem for hydrodynamics in the presence of high-explosive drive [8]
- Documentation of V&V assessments for this milestone: Summary of supporting documents:
  - Code verification assessments contained in this report relating to hydrodynamics [Section IV], radiative shocks [Section V], and materials [Section VI] capabilities within the LANL codes FLAG and xRAGE.
  - Presentation slides from Level 2 milestone review [9]

The infrastructure supporting code verification activities includes: ExactPack, LSPDM, and EOSlib, and the Verification Test Suite. ExactPack is a Python package that provides implementations of exact (analytic and semi-analytic) solutions to many commonly used code verification test problems. ExactPack is available as open-source software at http://github.com/lanl/exactpack. LSPDM (Lightweight Simulation Process and Data Management) is a Python package to manage the pedigree of simulation data and results. It is used to manage many of the code verification assessments described in this report. EOSlib is a C++ library used to provide reference implementations of analytic equations of state and reactive high explosive models. This library has existed with a limited user base at LANL for more than ten years. In our current work, we modernized the code and released it as open-source at http://github.com/lanl/eoslib.

The Verification Test Suite is a set of code input decks built to a standardized problem specification, with calls to ExactPack solvers and (if needed) EOSlib implementations, and stored in a repository that is managed by LSPDM (currently only for some problems, but eventually for all problems). The suite provides a reliable, repeatable way to run a particular test problem in a computational physics code. Changes in the results due to changes in the code or selection of different code options can be tracked and analyzed.

Current contents of the code verification suite cover the areas of hydrodynamics, high explosives, radiative shocks, and material models. The detailed list of which problems are implemented in which codes can be seen in the table of contents of this report. Four journal articles (cited above) have been produced documenting exact solutions and analyses of some of these problems.

Code verification assessments have been conducted and documented for the problems currently in the Verification Test Suite. Those results are documented in this report and in the presentation slides cited above. In the code verification assessments, we are looking to identify potential programming mistakes, as well as characterize algorithm behaviors. The main tool used for this is error convergence analysis. The convergence order for the codes analyzed is expected to be $p = 2$, $p = 1$, or $p = 2/3$, depending on the problem. We are also looking at some specific questions about our codes xRAGE and FLAG: 1) How does the new, unsplit hydro in xRAGE compare with existing algorithms? 2) How do different viscosity models affect results in FLAG? 3) How does ALE affect FLAG results? 4) What can we say about multi-physics models?

Overall, both xRAGE and FLAG exceeded our expectations for performance on the code verification problems of interest. We did note a number of items for follow-up and future work.

II. Code Verification Theory and Methods

A. Introduction to Code Verification

This report contains code verification assessments. To understand the meaning of a code verification assessment, we must first define a few terms. The definitions and methods used for code verification can be found in a few fundamental references from the field. The first two are National Standards produced by the American Society of Mechanical Engineers (ASME): *ASME V&V 10-2006: Guide for Verification and Validation in Computational Solid Mechanics* [10] and *ASME V&V 20-2009: Standard for Verification and Validation in Computational Fluid Dynamics and Heat Transfer* [11]. The third fundamental reference is the book *Verification and Validation in Scientific Computing* [12] by Oberkampf and Roy. These published references provide foundation of rigor for code verification theory and methods.

From Ref [10], we recognize two fundamental types of verification for scientific computing: Code verification and solution (or calculation) verification. The goal of code verification is to establish, through rigorous convergence analysis, that a computational physics code correctly implements a set of physics models and algorithms. In contrast, the goal of solution verification is to evaluate and analyze the uncertainties of code outputs associated with the choice of temporal and spatial discretization. This milestone and report are focusing only on code verification.

For code verification, we can list five types of verification criteria in increasing order of rigor [12]:

1) Simple tests: Symmetry, conservation, Galilean invariance
2) Code-to-code comparisons: Typically against a trusted reference code
3) Discretization error quantification: Compute error norm between code output and known solution
4) Convergence tests: Does code converge to known solution under refinement?
5) Order of accuracy tests: Does code converge to known solution at expected theoretical rate?

The first two are less rigorous, but can be applied to a wide variety of test cases with few limitations on problem complexity. The last three are more rigorous, but require a "correct answer" to use as a reference solution. For the order of accuracy tests, we must have access to two specific pieces of information: 1) What is the formal order of accuracy of the method? This comes from the theoretical calculation of truncation error between the discrete equations and the original partial differential equation (PDE); and 2) What is the observed order of accuracy of the computational physics code? This is the observed rate at which the discrete solution approaches the exact solution as discretization parameters go to zero. It is estimated using a global error norm between the computed and exact solutions, observed when systematically refining the computational mesh. This comparison of formal and observed orders of accuracy is the key aspect of code verification.

B. Error Taxonomy

In order to more fully understand how convergence analysis works and why it is such a valuable tool, it is helpful to examine the different errors that arise from numerical simulations. There are many different ways to categorize these, so we choose one that covers the categories of particular interest in the context of this report.

$$\varepsilon_T = \varepsilon_\Delta + \varepsilon_P + \varepsilon_I + \varepsilon_{RO} + \varepsilon_M \tag{1}$$

Eq. 1 lists some of the different errors that are encountered in the verification of codes:

- ε_Δ includes all simulation errors that are ordered in some measure of the problem discretization. Particularly, these typically include errors that result from the resolution of the spatial grid and the time step scheme used to obtain the simulation result.
- ε_P includes all simulation errors that are ordered in some physical parameter. Classic examples include Reynolds number (measuring the error associated with neglecting viscous effects), Mach number (measuring the error associated with ignoring the effects of material compressibility), or material strain (measuring the error associating with the small-strain approximation).
- ε_I includes all simulation errors associated with iterative convergence tolerances.
- ε_{RO} includes all simulation errors associated with round-off error.
- ε_M includes errors associated with any programming mistakes in the simulation code.

A fundamental assumption underlying the use of convergence analyses to verify simulation codes is that, under normal conditions, the error associated with discretization should be dominant, and the other terms should be negligible. Since only discretization error reduces as the grid is refined, any larger-than-expected "other" errors will manifest as a zeroth-order convergence rate.

C. Discretization Errors in 2nd-order, shock-capturing codes

The preceeding discussion was quite general, and it applies equally well to many different simulation codes. If we restrict our attention to 2nd-order, shock capturing codes, then the spatial discretization error term may be written:

$$\varepsilon_\Delta = a(\Delta x)^{2/3} + b\,(\Delta x) + c(\Delta x)^2 + f\,(\Delta t) \tag{2}$$

The quadratic term in Eq. 2 represents errors inherent in smooth regions, while the linear term represents errors arising from the action of limiters near non-linear waves (rarefactions and shocks). The $2/3$ term is a result of the particular errors that arise from linearly degenerate waves such as contact discontinuities [13]. The particular values of the coefficients a, b, and c depend on the details of the algorithm implemented in the code, as well as the particular test problem being modeled.

Given this particular error model, we expect to see an overall convergence rate for our test problems of 2, 1, or 2/3, depending on the features included in a particular test problem, at sufficiently fine resolutions. For xRAGE modeling a problem that includes a contact discontinuity (e.g. a Riemann problem), we would then expect to see 2/3-order convergence. For FLAG modeling the same problem in Lagrangian mode (i.e. without ALE), we would expect to see 1st-order.

D. Use of Exact Solution Problems

For rigorous verification (Levels 3-5), a known solution to the underlying governing equations is required. Using test problems where an exact solution is known is the most straightforward way to accomplish this. The exact solutions can be analytic (i.e. closed-form algebra) or semi-analytic (i.e. require solution to a transcendental equation or an ordinary differential equation). These problems are a good starting point for code verification as they are easy to understand and computationally straightforward to solve. They are also typically easy to implement in the computational physics code. However, such problems are limited in dimensionality, complexity of physics, etc. Additional approaches that preserve rigor while increasing complexity include the Method of Manufactured Solutions (MMS) [12], which is beyond the scope of the current milestone work but will be explored by the project team in the near future.

E. Use of Error Norms and Convergence Plots

A plot of the convergence of the error norm is a useful tool in code verification. It allows identification of the asymptotic region of convergence, the rate of which is the observed order of convergence. The quantities of interest for code verification are the zonal physical variables (density, pressure, and specific internal energy), and the nodal velocity. The verification error metric will be the weighted L_1 error norm between the exact and calculated solution values for each quantity of interest:

$$\epsilon_n = \frac{\sum_{i=1}^{N_Z} w_i \left| f_i^{calc} - f_i^{exact} \right|}{\sum_{i=1}^{N_Z} w_i} \tag{3}$$

where f_i^{calc} is the quantity of interest as calculated by the computational physics code at zone i, and f_i^{exact} is the exact solution of the same quantity at the same location. For zonal quantities, w_i is the length/area/volume of the zone, and for nodal quantities, $w_i = 1$.

A key element of the verification analysis is the calculation of the convergence rate of the L_1 error norm as a function of the initial grid length h_n. The standard error *ansatz* for verification analysis is assumed:

$$\vec{\epsilon} = A\vec{h}^{p_{obs}} \tag{4}$$

where $\vec{\epsilon}$ is the vector of errors ϵ_n as computed using Eq. 3 and $\vec{h}$ is the vector of initial zone length values h_n. The observed order of accuracy p_{obs} is determined from Eq. 4 using linear regression in log space:

$$\log \vec{\epsilon} = \log A + p_{obs} \log \vec{h} \tag{5}$$

Plotting the error norm ϵ_n for each grid n for each method to be compared results in a plot of the type shown in Figure 1. We can visually identify the linear region between $\delta x = 4e^{-4}$ and $\delta x = 3e^{-3}$ as being the region of asympotic convergence, i.e. the region where the lowest order terms of the error expansion are dominant. Identification of the asymptotic region is important to understand the error behavior of the algorithm as implemented. Also note that the slopes of the various lines fit in the plot appear in the legend. The slope of the fit in the region of asymptotic convergence is the observed order of convergence (e.g. $p = 0.963$ for the red line). The cor value is the correlation of the fit with the data points, where $cor = 1.000$ indicates a perfect fit to three significant digits. It should be noted that most verification test problems are scale invariant; so the scale and units of the problem are chosen arbitrarily, so users of the codes should not attribute meaning to the absolute magnitude of the zone sizes appearing in this report.

Now with Figure 1 in mind, we recall the verification criteria listed above: Level 3 verification requires the quantification of error between the computed solution and the known solution, the data points appearing on this plot represent that level of rigor. Level 4 verification require observation of convergence, which simply means that the magnitude of the error points is decreasing as the mesh becomes refined (i.e. we move to the left on this plot). Level 5 verification requires the fitting of a convergence rate, which has been completed, with the lines representing the fits.

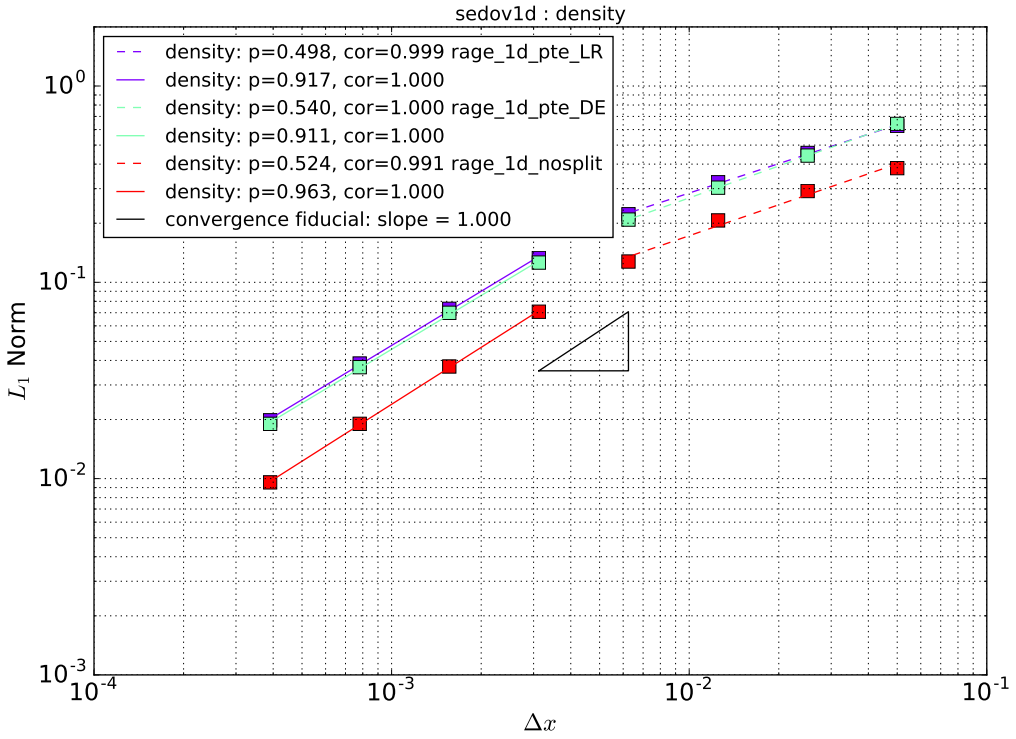


Fig. 1: Sample convergence plot to illustrate the process of code verification

F. The Verification Test Suite

To cover a broad variety of the capabilities in a computational physics code, a suite of verification test problems is required. A Verification Test Suite requires the following components. For each component, we indicate what aspect of the milestone satisfies and documents this requirement.

- Standardized definitions for verification test problems: We have published a set of such standardized definition [4], which we colloquially call the one-pagers. These definitions are intended to specify the problem setup sufficiently to allow implementation in any code. Note that as a result of attempting to be "code agnostic," it is often the case that the problem is not implemented optimally in a particular code.
- Test problem exact solutions: Our code *ExactPack* calculates these. This is discussed more in the following section.
- Input deck generation and simulation execution. This is managed by our code *LSPDM*. This is discussed more in the following section.
- Code verification analysis: This includes the error convergence analysis described above. *ExactPack* provides tools to perform this analysis.
- Documentation and archiving: This is also managed by *LSPDM*.

III. Physics Verification & Validation Infrastructure

A. Introduction: Overview of the Software Solution Provided

For verification and validation of large-scale, multi-physics codes two requirements have guided our software development: Results must be *pedigreed* and *shareable*. To solve the V&V problem for ASC codes we have developed four software tools:

- ExactPack
- LSPDM
- PhysVal CMF (described in a separate report)
- EOSlib

ExactPack is a Python package which provides analytic or semi-anlytic solutions to test problems relevant to the physics in ASC simulations. To manage test problems that can be formulated and solved with several different equations of state we have adopted EOSlib. The PhysVal Common Model Framework (CMF) has been built to ensure reproducibility and accurate communication of computational modeling choices made for simulating validation experiments. Simulations and analysis created using these tools can be tracked and executed through our Lightweight Simulation Process and Data Management (LSPDM) tool. Finally, these tools can be combined to track verification and validation simulation results, the simulation set-up and execution environment, and the analysis of simulation results. One example is the Verification Test Suite (VTS), which packages a set of test problems as an LSPDM repository using ExactPack.

All of our development is done using modern best practices for software engineering.

B. Lightweight, Simulation, Process, and Data Management (LSPDM)

In many large engineering companies reproducibility and archiving of a series of simulation studies for a production system is necessary for the firm to provide confidence and accountability to their customers. In the weapons program we also strive to provide a high level of confidence in our simulation ability to the U.S. government. To deliver this our code verification suites must be immediately reproducible to aid in communication to independent evaluators. In industry, archiving of studies is commonly being handled by *Simulation Process and Data Management* (SPDM) software. An SPDM's role is to allow the user to enter a particular study to be run, specify the code that will run the study, direct where study results are to be stored, assign a computing platform to execute the code, and then execute and store the results of analysis. This is often provided in a GUI framework.

For the verification and validation of physics software the corresponding requirement is tracking data generated from a suite of verification or validation studies. This involves recording the exact process that was actually run on the computer to generate the data, the software environment, and the post-processing that was performed to draw conclusions from the data. Ideally, one team should be able to set-up and run a series of simulations for a convergence or parameter study, perform analysis on the data, and have a second team be able to checkout the study from a repository and run the exact same processes. To solve this problem the Physics Verification team has adopted the SPDM approach.

Due to the complexity of the task that an SPDM must perform the software is usually set up by an outside firm for a client company. The set up on an SPDM system is then only for the specific codes that the client wishes to run along with the specific computer architecture and analysis software that the client company uses. In the ASC program our codes, analysis scripts, run-time environment, and HPC system changes fast enough to make third party management of an SPDM a significant source of additional overhead. Additionally, our security posture is a challenging environment for third-party vendors to support. For this reason our team developed an in-house version of SPDM. Our solution is dubbed *Lightweight Simulation Process and Data Management* (LSPDM).

LSPDM is implemented as a Python package that acts as a frontend to a Subversion (SVN) repository. The module enables a command-line interface capable of creating a repository, checking out a working copy of the repository, specifying the computational environment of processes in the repository, and adding simulation and analysis processes to the repository. LSPDM itself is maintained using our software engineering standards (see III-F).

Complete documentation for LSPDM, including a User's Guide and a Developer's Guide, can be found in the LSPDM manual.

C. ExactPack

The largest single piece of software developed for the Physics Verification project is *ExactPack*. ExactPack is a Python package providing analytic or semi-analytic solutions of a host of physics problems, all with a common API. The primary purpose of ExactPack is produce reference solutions for verifying the behavior of large-scale physics codes such as FLAG and xRAGE. ExactPack also contains tools to do verification analysis tasks. This is a sizeable software undertaking, involving contributions of up to 10 developers and subject matter experts, and the project requires the use of modern software engineering tools (see III-F).

Although most problems of interest do not have analytic or semi-analytic solutions, there are a very large number of relevant, nearby problems that do. A small number of those are closed-form analytic expressions, but most require an integral quadrature

or ODE solve. Computing these semi-analytic solutions is often quite challenging and requires expertise both in the physics subject area and in numerical methods. As a result, there are many different codes available for various cases, each with its own setup and output formats. The overhead in collecting, deploying, maintaining, and interfacing with each individual code creates a substantial time requirement in verification.

ExactPack is intended to simplify this process by integrating carefully vetted solvers for many different physics problems into one common interface. The architecture is designed so it is easy to create new solvers, or to integrate other existing standalone codes. Since the output format is standardized, ExactPack can also use standard tools to perform analysis. Leveraging VTK for reading code data means that ExactPack can understand a wide variety of file formats. Data in any compatible format can be compared to output from any of the included solvers using only a few lines of Python code. Convergence analysis and standardized plotting is also extremely easy.

Currently ExactPack includes solvers in five different broad physics areas, each including multiple specific solvers:

- Pure Hydrodynamics
  - Riemann Shock Tubes (6 problem variants): Sod, Einfeldt, Stationary-Contact, Slow-Shock, Shock-Contact-Shock, LeBlanc
  - Riemann with the JWL EOS (2 variants)
  - Noh
  - Nohs second problem (uniform collapse)
  - Sedov
  - Guderley
- Conduction
  - Heat conduction problems
  - Coggeshall problems (20 of the 22 problems)
  - Reinicke Meyer-ter-Vehn
  - Su-Olson
- Radiation Hydrodynamics
  - Lowrie-Rauenzahn equilibrium diffusion
  - Lowrie-Edwards non-equilibrium diffusion
  - Ferguson-Lowrie-Morel grey Sn transport
- High Explosives
  - Mader
  - Escape of High Explosives Problem
  - Steady Domain Reaction Zone Problem
  - Kenamond/Programmed Burn
  - Detonation Shock Dymamics
  - Steady Zeldovich-von Neumann-Dring (ZND) wave tests for arbitrary reaction rate law and equation of state (coming soon)
- Materials
  - Blake
  - Hunter

Users can have confidence in reference solutions produced by ExactPack because the library includes over 450 well-documented tests which all releases must pass. Complete documentation for ExactPack, including a User's Guide, a Developer's Guide, and a complete description of all the solvers and internal testing, can be found in the ExactPack manual.

D. EOSlib

EOSlib is a software package for thermodynamic computations involving analytic equations of state. It was developed by Ralph Menikoff, and has been repackaged by the verification team to build and install cleanly on a wide variety of platforms. It is intended to be used as a single, standard, equation-of-state package within ExactPack, in order to simplify the process of creating exact solvers for a wider variety of EOS models.

The principal benefits of developing a stand-alone EOS package are:

- A single target for verification efforts involving thermodynamic models.
- Easy access to different thermodynamic models for ExactPack solvers.
- Availability of a reference implementation of thermodynamic models for the purposes of code-to-code comparisons.

EOSlib currently contains implementations of a variety of EOS models, principally focused on HE applications. Some of these include: Ideal Gas; Stiffened Gas; Hayes (including extensions); JWL; Davis. EOSlib can also handle mixtures, through a pressure-temperature closure model, as well as quasi-equilibrium models such as reactive burn and material elasticity/plasticity.

New materials are easily added to the pre-packaged data files, and the design of the software is well-suited to computing environments where some data must be segregated and protected differently.

Documentation is currently available as part of an old report [14], and EOSlib has also been used as part of a recent verification study on the ZND wave [15]–[17].

Current work on EOSlib is focused on a major code refactor, as part of the development of a Python-based interface to the underlying C++ code. Early prototypes of this interface have been used to implement a general ZND solver in ExactPack.

EOSlib has been released as an open-source software package, available at https://github.com/lanl/EOSlib. The internal repository is located on BitBucket, at https://xcp-stash.lanl.gov/projects/PHYSVER/repos/eoslib/browse. Feedback and contributions are greatly appreciated.

E. The Verification Test Suite

The LSPDM and ExactPack software tools provide the ability to create version controlled verification studies, involving a large suite of test problems, for several ASC codes. These codes include FLAG, xRAGE, and Pagosa. We have created LSPDM repositories for most of the standardized test problem definitions [4] for at least one code, with several for two or three codes. These repositories are collectively called the *LANL Verification Test Suite*, or VTS.

The VTS establishes conventions for directory organization for test problems to facilitate sharing repositories and for performing parameter studies. The intention is to make it easy to run a problem, or set of problems, with a new version of the code, or a different set of code settings. VTS repositories are designed to run in the LANL HPC environment, and use the HPC modulefiles capability to track the environment used for running codes inside the suite.

The basic building blocks of the VTS are resolutions studies. Each resolution study consists of a input file template which can be used to generate input files for each resolution via an LSPDM process. Each resolution lives in a separate directory, and is also run using an LSPDM process. A master postprocessing LSPDM process compiles the results into the desired plots. The entire chain can be run with a single LSPDM command. A detailed description of how such a workflow can be created (for an example code) can be found in the LSPDM manual [2, sec. 2.6].

Once a resolution study has been created for a problem, creating modified version for similar cases, such as running with a different artificial viscosity model or setting up one Reimann problem from another, is simple. The user simply uses LSPDM to make a copy of the first case, and makes the appropriate modifications to the input templates in the copied version. The same LSPDM command can then be used to rerun the solvers and update all the data in the copied case to correspond to the new case. By repeating and nesting this process, one can easily build up the complex hierarchies of cases that make up the full VTS.

F. Software Engineering

1) Version Control: All of our codes use Distributed Version Control (DVCS) to increase productivity, coordinate work within the development team and with outside collaborators, keep a historical record of code development, and distribute code products. The DVCS we use is Git, and we maintain two Git servers for our codes: one an internal LANL server (Atlassian BitBucket) managed by WRS-SNA and another on GitHub (www.github.com/lanl).

DVCS allows developers to do extensive development work on their own repository copy, with frequent multiple commits, but still not share their changes until the code is sufficiently mature. We use a branched development model, in which each feature or bugfix is developed in its own branch, and is merged into the master development branch only after a *pull-request*. Pull-requests are not approved until the code has been reviewed and found to meet all our projects development standards and requirements, as well as pass all the tests.

External collaboration can be done on the same model, using project forks via GitHub, rather than branches in our internal BitBucket server.

2) Bug Tracking: For bug tracking, task management, and code roadmap, we use JIRA, a proprietary ticketing system from Atlassian which is managed locally by WRS-SNA. JIRA allows both users and developers to submit tickets for bugs, improvements, and feature requests. Tickets are assigned to members of the development team, and progress can be tracked and status of particular tasks can be monitored by other team members. Wish list features can be left as unassigned tickets until resources are available to work on them.

3) Documentation: For documentation, we use Sphinx, which is a widely used documentation generator, particularly in the Python community. (Python's own documentation is written in Sphinx.) Sphinx documentation is prepared using ReStructured Text (ReST) markup language, which is designed to be human readable and editable. It produces output in many formats; we release our documentation in HTML on-line as well as in PDF printable versions.

Sphinx can also extract API documentation from Python docstrings (which are the standard commenting protocol for Python code). We have strict internal requirements that code must be properly documented before it can be merged into the master development branch. This assures that the API section of our documentation is always comprehensive and up-to-date. We also maintain additional sections which describe how to use the codes, our development strategy and coding requirements, and careful descriptions of the code physics and numerical methods.

4) Testing: There are several testing systems available for Python. The most widely used is probably `unittest`, which is available as part of the standard Python library. All of our tools use `unittest` for both unit and integrated testing. Passing the entire unit test suite is a requirement for any committed code, as is a sufficient level of test coverage for the new code.

One open issue is how to test verification codes which are intended themselves for testing other codes. Our approach has been to compare to well vetted solutions, either from historical codes with a long pedigree, or from well vetted literature. Comparisons are documented so that users can decide for themselves on the confidence level they provide.

There are also many tests to check that the API performs according to specifications: that methods do the correct argument type checking, errors are raised when necessary, and similar code infrastructure tests.

5) Continuous Integration: Continuous integration is the automation of regularly testing a code repository. Scheduled tests are performed at specified intervals, nightly, or whenever code is merged into the master repository from a pull-request. Jenkins is a web-applet that can be modified to know about projects under version control within the Atlassian tool suite and perform continuous integration. Continuous integration with Jenkins works as follows. A Jenkins task is set up to pull source from a code repository. After this, Jenkins executes a script, under version control, that installs the software and runs the test suite. Test results are then reported through the Jenkins web-applet. If the tests are all successful the build is declared successful, if some tests fail then the build is declared to fail. Jenkins also processes the testing data to determine *coverage*, how much of the source is actually tested. This shows the development team where more testing is needed. The task just described is performed nightly by Jenkins and whenever there is a pull-request issued.

IV. Verification of Hydrodynamics

A. Description of the Problems

1) The Continuum Equations of Motion: The state of a fluid at any given time can be described by six functions of space: the pressure, the mass density, the specific internal energy (energy per unit mass), and the fluid velocity (one for each spatial direction),

$$P(\mathbf{x}), \quad \rho(\mathbf{x}), \quad e(\mathbf{x}), \quad \mathbf{u}(\mathbf{x}) \ . \tag{6}$$

The three of variables P, ρ, and e are not independent, and any one can be written as a function of the other two via an equation of state (EOS), which we write as

$$P = P_{EOS}(\rho, e) \ . \tag{7}$$

The Euler equations express the conservation of mass, momentum, and energy, and take the general form

$$\frac{\partial \rho}{\partial t} + \boldsymbol{\nabla} \cdot \Big(\rho \, \mathbf{u} \Big) = 0 \tag{8}$$

$$\frac{\partial \rho \, \mathbf{u}}{\partial t} + \boldsymbol{\nabla} \cdot \Big(\rho \, \mathbf{u}\mathbf{u} \Big) + \boldsymbol{\nabla} P = 0 \tag{9}$$

$$\frac{\partial}{\partial t} \left(\frac{1}{2} \, \rho u^2 + \rho \, e \right) + \boldsymbol{\nabla} \cdot \left(\frac{1}{2} \, \rho u^2 + \rho \, e \right) \mathbf{u} = 0 \ . \tag{10}$$

The 5 Euler equations, plus the EOS, give a total of 6 constraint equations (7)-(10) for the 6 unknown state variables in (6); therefore, the Euler equations have a unique mathematical solution (given appropriate boundary and initial conditions).

The simplest way to find exaction solutions for code verification is to simplify the problem to a polytropic ideal EOS, in which (7) takes the form,

$$P = (\gamma - 1) \rho e \ , \tag{11}$$

where $\gamma \equiv c_p / c_v$ is the ratio of specific heats at constant pressure and volume. We then look for spherically symmetric solutions to the Euler equations, which renders the variables functions of the radial coordinate $r = |\mathbf{x}|$. The independent fluid variables are then, (i) the gas density $\rho(r,t)$, (ii) the radial velocity of the gas $u(r,t)$, and (iii) the pressure $P(r,t)$, each at spatial location r and time t. Since $u(r,t)$ is the radial velocity, a negative value indicates that gas is flowing in toward the origin. In fact, we can generalize this symmetry argument, and express the Euler equations not only in spherical geometry, but also in cylindrical and planar geometries (distinguished by the *geometry parameter* $k = 1, 2, 3$ respectively). In the case of planar coordinates, the variable r is just the x variable, and for cylindrical coordinates, r is the polar variable in the x-y plane. The Euler equations now take the simpler form

$$\frac{\partial \rho}{\partial t} + u \, \frac{\partial \rho}{\partial r} + \frac{\rho}{r^{k-1}} \frac{\partial}{\partial r} \Big(u \, r^{k-1} \Big) = 0 \tag{12}$$

$$\frac{\partial u}{\partial t} + u \, \frac{\partial u}{\partial r} + \frac{1}{\rho} \frac{\partial P}{\partial r} = 0 \tag{13}$$

$$\frac{\partial}{\partial t} \Big(P \rho^{-\gamma} \Big) + u \, \frac{\partial}{\partial r} \Big(P \rho^{-\gamma} \Big) = 0 \ , \tag{14}$$

where $k = 1, 2, 3$ is the geometry factor specifying planar, cylindrical, or spherical symmetry, respectively. In the energy equation, we have used an entropy conservation condition.

The hydro suite consists of the Noh and Sedov problems and six variants of the Riemann problem. There are many definitions of these problems in the literature, and we use the definitions in Ref. [4]. The Noh and Sedov solutions are normally spherical (there are cylindrical and planar counterparts with $k = 2, 3$, although we shall not consider them here), while the Riemann shock tube problems are 1D planar with $k = 1$.

2) Noh Problem: At the initial time, the Noh problem is defined by a gas moving radially inward with uniform constant velocity $u(r, 0) = -1 \, \mathrm{cm/s}$, zero pressure $P(r, 0) = 0$, and constant density $\rho(r, 0) = 1 \, \mathrm{g/cm^3}$. An infinitesimal time later, an infinitely strong stagnation shock forms at the origin $r = 0$ and starts propagating outwards, leaving behind a region of non-zero pressure and internal energy in its wake. The Noh problem exercises the code's ability to transform kinetic energy into internal energy, and is sensitive to the fidelity with which supersonic flows are tracked. The analytic solution is particularly simple, and takes the closed form given by Eqs. (15)-(25) below. Working in dimensionless units (in which the velocity $u_0 = 1 \, \mathrm{cm/s}$ and mass density $\rho_0 = 1 \, \mathrm{g/cm^3}$ have been scaled out), the outgoing shock position is given by

$$r_{\mathrm{shock}} = \frac{t}{3} \ , \tag{15}$$

with

$$r < r_{\rm shock}(t): \tag{16}$$

$$\rho(r,t) = \left(\frac{\gamma+1}{\gamma-1}\right)^{k+1} \tag{17}$$

$$e(r,t) = \frac{1}{2} \tag{18}$$

$$u(r,t) = 0 \tag{19}$$

$$P(r,t) = \frac{4^{k+1}}{3} \tag{20}$$

behind the outgoing shock, and

$$r > r_{\rm shock}(t): \tag{21}$$

$$\rho(r,t) = \left(1+\frac{t}{x}\right)^{k-1} \tag{22}$$

$$e(r,t) = 0 \tag{23}$$

$$u(r,t) = -1 \tag{24}$$

$$P(r,t) = 0 \tag{25}$$

in front of the shock. For generality, we have kept the geometry factor $k = 1, 2, 3$ explicitly, which corresponds to planar, cylindrically, and spherical geometries, respectively. The test suite only employs the spherical case with $k = 3$. The correct dimensions can be restored by inserting appropriate factors of $u_0 = 1\,\mathrm{cm/s}$ and $\rho_0 = 1\,\mathrm{g/cm^3}$, for which the shock position becomes

$$r_{\rm shock}(t) = \frac{1}{3}\,u_0\,t\ . \tag{26}$$

For the spherical case ($k = 3$), the solution takes the form shown in Fig. 2.

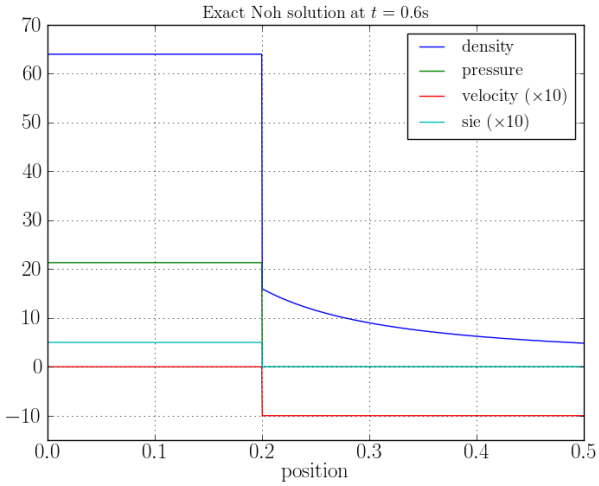


Fig. 2: The exact spherical Noh solution at $t = 0.6\,\mathrm{s}$ with $\gamma = 5/3$.

3) Sedov: The (spherical) Sedov blast wave is a mathematical idealization of a point-source explosion in which energy E_0 is released instantaneously from the origin ($r = 0$). By convention we choose the the initial time (at which the energy is released) to be $t = 0$. The shock wave occurs in an ambient thermodynamic fluid medium, such as air (we take the EOS of this medium to be that of an ideal fluid with gas parameter γ, mass density ρ, and specific energy e). The resulting explosion forms an infinitely strong outward moving shock wave at $t = 0$. For simplicity, the initial density $\rho(r, 0)$ is taken to a constant, which is often chosen to be $\rho_0 = 1\,\mathrm{g/cm^3}$, while the initial velocity and pressure are taken to be zero everywhere ($r > 0$), $u(r, 0) = 0$ and $P(r, 0) = 0$. The total energy of the gas is

$$E_0 = \int dV \left[\frac{1}{2}\,\rho u^2 + \frac{P}{\gamma-1}\right]\ , \tag{27}$$

where the integration runs over the volume of the shock at time t, *i.e.* the integration runs over $0 \leq r \leq r_{\rm shock}(t)$ [1]. Upon choosing $\gamma = 7/5 = 1.4$ and energy $E_0 = 0.851072\,{\rm erg}$ (with constant initial density $\rho_0 = 1\,{\rm g/cm^3}$), the shock will arrive at $r_{\rm shock} = 1\,{\rm cm}$ at time $t = 1\,{\rm s}$, as illustrated in Fig. 3.

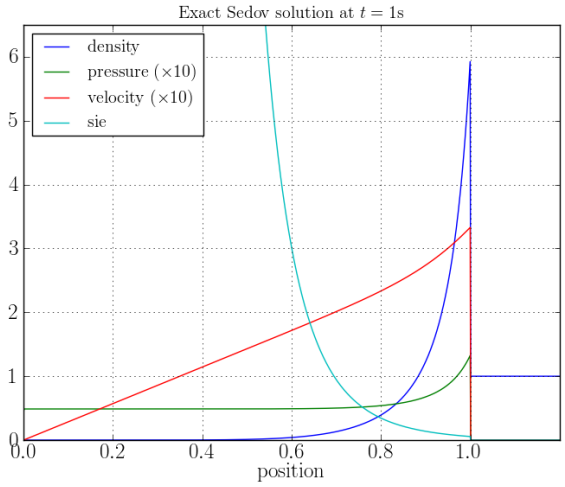


Fig. 3: The exact spherical Sedov solution at $t = 1.0\,{\rm s}$ with $\gamma = 7/5 = 1.4$.

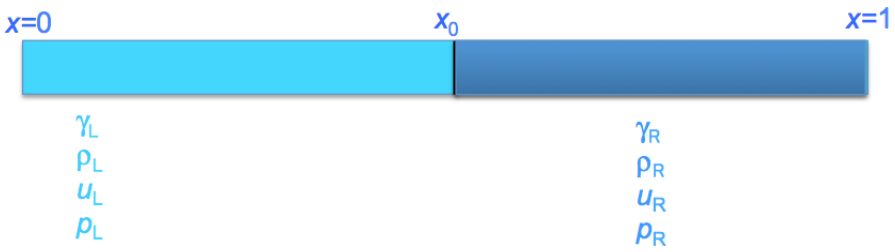


Fig. 4: The Riemann shock tube.

4) Riemann: Riemann problems are mathematically idealized shock tube experiments in Cartesian (or planar) geometry. The initial setup is illustrated in Fig. 4, in which two gasses are separated by a membrane at $x = x_0$ By convention, the left end of the tube is at $x = 0$ and the right at $x = 1\,{\rm cm}$, and the gas in the tube is divided into left and right components by a membrane at x_0 with $0 < x_0 < 1$. The gas on the left has mass density ρ_L, pressure p_L, fluid velocity u_L, and gas constant γ_L. The state of the gas to the right of the membrane is specified by similar quantities, ρ_R, p_R, u_R, and γ_R. In principle, the gas constants may differ, but in this suite $\gamma_L = \gamma_R$. The initial state is therefore given by

$$x < x_0: \tag{28}$$
$$\rho(x,0) = \rho_L \tag{29}$$
$$u(x,0) = u_L \tag{30}$$
$$p(x,0) = p_L \tag{31}$$

and

$$x > x_0: \tag{32}$$
$$\rho(x,0) = \rho_R \tag{33}$$
$$u(x,0) = u_R \tag{34}$$
$$p(x,0) = p_R\,. \tag{35}$$

The membrane is removed at the initial time $t = 0$, and the gas dynamics determine the subsequent state at any later time t. The state will always be a combination of shocks (discontinuities in all state variables), contact discontinuities (discontinuities in ρ but not p and u), and rarefaction waves.

[1] The volume element takes the form $dV = dr$ for $k = 1$, and $dV = 2\pi r\,dr$ for $k = 2$, and $4\pi r^2\,dr$ for $k = 3$.

TABLE I: Parameters for the six Riemann test problems.

Test	x_0 cm	t_{fin} s	γ [-]	ρ_L g/cm^3	u_L cm/s	p_L dyn/cm^2	ρ_R g/cm^3	u_R cm/s	p_R dyn/cm^2
1	0.5	0.25	7/5	1.0	0.0	1.0	0.125	0.0	0.1
2	0.5	0.15	7/5	1.0	-2.0	0.4	1.0	2.0	0.4
3	0.8	0.012	7/5	1.0	-19.59745	10^3	1.0	-19.59745	10^{-2}
4	0.5	1.0	7/5	3.857143	-0.810631	10.33333	1.0	-3.44	1.0
5	0.5	0.3	7/5	1.0	0.5	1.0	1.25	-0.5	1.0
6	0.3	0.5	5/3	1.0	0.0	$(2/3) \times 10^{-1}$	10^{-2}	0.0	$(2/3) \times 10^{-10}$

For the suite, we implement six variants of the Riemann shock tube, as defined in Table I, and summarized in the following [18]:

1. *Sod shock tube*. This is the canonical shock tube problem with rarefaction-contact-shock structure. This is not computationally challenging, but quickly identifies algorithmic problems resolving basic wave structure.

2. *Einfeldt (or 1-2-3) problem*. This problem consists of two strong rarefaction waves with a near-vacuum between them. Methods that conserve total energy might show internal energy errors.

3. *Stationary contact problem*. This problem consists of a strong shock wave moving to the right, a stationary contact, and a strong rarefaction wave moving to the left. This problem tests an algorithm's dissipation by how much the contact is smeared.

4. *Slow shock problem*. This problem consists of a Mach 3 shock wave moving slowly to the right. Some numerical methods exhibit unphysical oscillations behind the shock.

5. *Shock-contact-shock problem*. In this problem, two shocks separate from the initial state with a contact between them. This produces errors in all the fields, and this problem tests an algorithm's ability to deal with these errors.

6. *LeBlanc problem*. This problem is a strong shock and strong rarefaction version of the basic rarefaction-contact-shock problem, and is a good test for the robustness of an algorithm.

The exact solutions of these problems are graphed below. Test1 is show in Fig. 5, while test2 or the Einfeldt problem is given by Fig. 6. Note the opposing rarefaction waves of test2 moving left and right, and thereby leaving a near-vacuum in the central region.

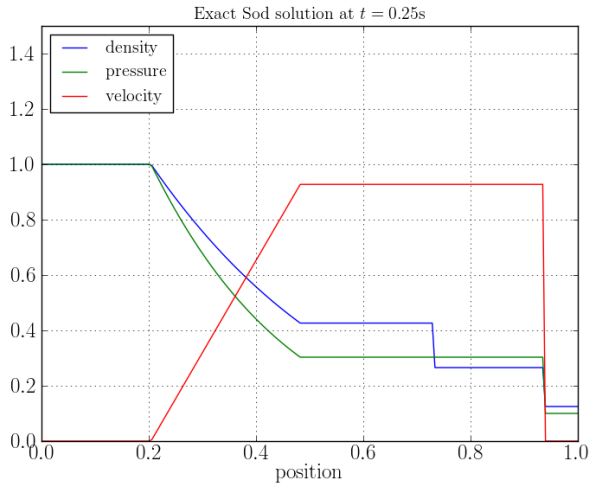


Fig. 5: The exact Sod solution at $t = 0.25\,\text{s}$

The stationary-contact of test3 in Fig. 7 is exhibited near $x \sim 0.8$.
The slow-shock of test4 in Fig. 8 has traversed a distance from $x \sim 0.5$ (the initial membrane location) to $x \sim 0.6$ in the time interval $\Delta t = 1\,\text{s}$.
The shock-contact-shock structure of test5 is illustrated very cleanly in Fig. 9, where the first shock is near $x \sim 0.2$ (discontinuities in density, pressure, and fluid velocity), the contact is near $x \sim 0.5$ (discontinuity in only density), and the final shock is near $x \sim 0.8$ (with discontinuities in all fluid quantities). Finally, Fig. 10 illustrates test6/LeBlanc.

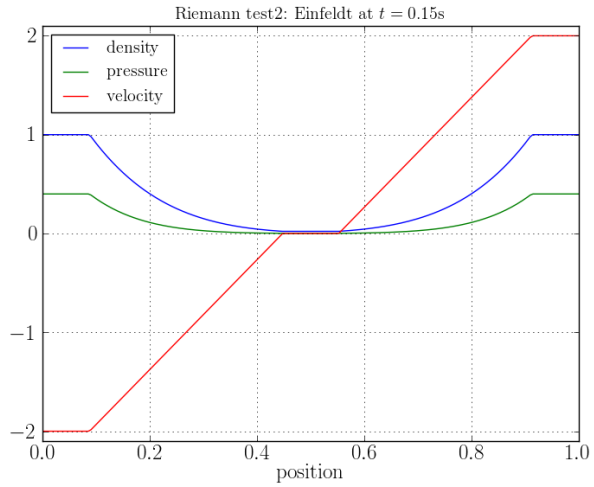


Fig. 6: Exact solution for test2/Einfeldt.

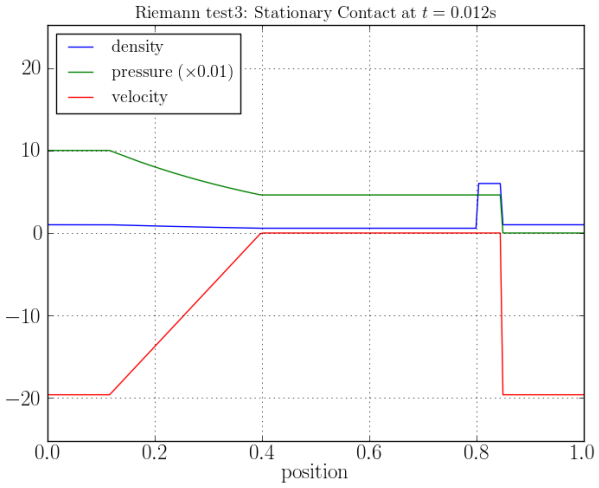


Fig. 7: Exact solution for test3/stationary-contact.

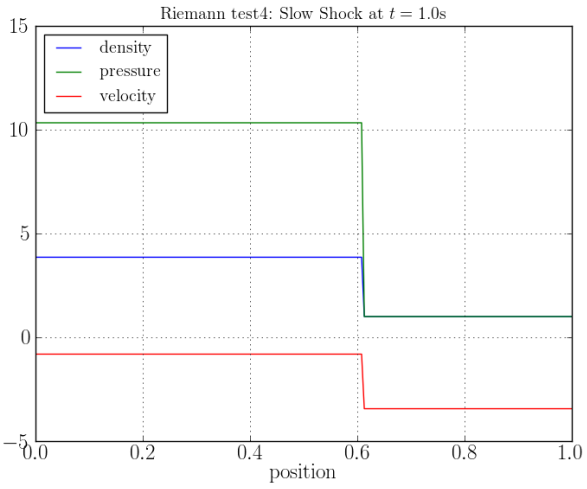


Fig. 8: Exact solution for test4/slow-shock.

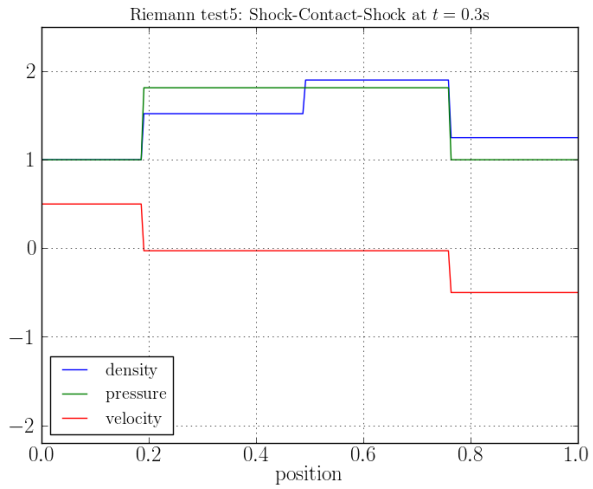


Fig. 9: Exact solution for test5/shock-contact-shock

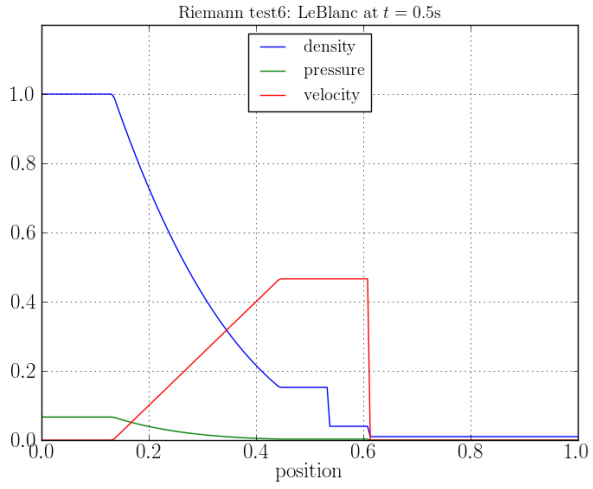


Fig. 10: Exact solution for test6/LeBlanc.

B. FLAG Hydrodynamics

This section presents a verification assessment of the hydrodynamics in the LANL multi-physics code FLAG. We are looking for potential programming mistakes, as well as to characterize the behavior of the algorithm, especially the convergence of the error. We are using the staggered-grid Lagrangian hydro algorithm in FLAG, which is formulated to be second-order accurate. Therefore, we expect $p = 2$ convergence for smooth problems, $p = 1$ convergence for problems with a discontinuity (including a shock), and $p = 2/3$ for problems containing a contact surface. For each problem, we will observe the error convergence, see if it is converging asymptotically, and if so, fit a convergence order to the error norm. We compare this value to the expected order. If the error norm converges at something other than the expected order, this is still good, but it does invite comment as to why the order shows that value. It should be noted that all of the code verification theory described previously assumes a fixed (Eulerian) grid, so we must exercise more judgment when analyzing the results for FLAG. In addition to the motivation provided above, we are looking to address some specific questions in this analysis, such as how different artificial viscosity models and the use of Arbitrary Lagrangian-Eulerian (ALE) methods affect the results in FLAG.

The artificial viscosity models to be tested include:

- *Barton* operates on a velocity jump/change along edges, i.e. differences in edge endpoint velocities. Thus it is called an "edge-based" method. It is aware of shock direction, but lacks limiters, and so is active everywhere in the mesh. It is known to work well when the shock aligns with the mesh, but can introduce mesh distortion otherwise. [19]
- *BBL* is a symmetric tensor formulation due to Brezzi, Buffa, and Lipnikov [20]. It is a reformulation of Campbell-Shashov tensor viscosity [21]. It applies zone-centered dissipation forces and has limiters, so that it should be inactive away from shocks
- MARS (Multidirectional Approximate Riemann Solver) [22] is a staggered-grid Godunov-like approach that uses an approximate Riemann solution to enable numerical solutions across shock discontinuities. This replaces the traditional artificial viscosity model, but we include it under the heading of "artificial viscosity models" for the purposes of this code verification analysis.
- VNR (Von Neumann Richtmeyer) is a classical quadratic artificial viscosity model with a Landshoff linear term.

1) Summary of Observations:

- Overall, FLAG met or exceeded our expectations for most of the verification tests that we ran.
- There are many questions surrounding the use of standard, Eulerian-based verification methods with Lagrangian/ALE codes.
- FLAG converges as expected for Riemann problems, with some unexplained behavior around the slow-moving shock.
- An appropriate artificial viscosity model must be chosen based on the needs of the user; MARS is very accurate for Lagrangian computations, while BBL is very robust, symmetric, and performs well with ALE.
- Specific test problems (Noh, Sedov), as set up for this analysis, are very challenging for edge-based viscosity methods (e.g. Barton). This trend does not persist when using ALE.

C. FLAG Sedov Problem

In FLAG, the Sedov problem grids are set up as specified in the project standard definition [4]. Specifically, the problem is set up with two materials; one the background gas and one the energy source. This actually complicates the solution, especially for ALE, and a good follow-on activity would be to set up a simpler version that only has one material. Note that for all of the FLAG Sedov comparisons, the convergence analysis is performed using the L_1 norm over the domain $[0.75, 1.2]$.

1) Summary of Observations:

- Barton viscosity works very well in 1D for Lagrangian, and works well in RZ and 3D in ALE mode
- In RZ and 3D, with pure Lagrangian, Barton viscosity, being an edge-based method, is very sensitive to the fact that the problem setup introduces the shock transverse to the mesh, which results in mesh distortion. Using a mesh that aligns better with the shock motion should result in better behavior.
- BBL is robust and symmetric in both RZ and 3D. It is, however, dissipative, which results in not capturing the full height of the shock and thus exhibiting overall lower accuracy compared to the MARS methods.
- For ALE with the Euler relaxer, the results are less sensitive to artificial viscosity models than with pure Lagrange, but are very sensitive to the details of the advection settings. Overall, the best results were observed when using code team recommended settings for advection plus turning on kinetic energy conservation.

2) Sedov in FLAG, 1D Spherical, Lagrangian: The specification of the problem setup for a general hydrodynamics code is shown in Ref [4]. The mesh setup for the Sedov problem in FLAG with 1D spherical geometry is straightforward. In the FLAG setup, we choose to define the single-zone energy source region as one material, and the remainder of the gas as another material. We use the Barton artificial viscosity model.

- Convergence of L_1 error norm:
  Running with 11 levels of resolution (from $nz = 24$ to $nz = 24576$) yields the convergence shown in Figure 11. By the 2nd refinement (96 zones), the error convergence is clearly in the asymptotic regime and converges thereafter at a rate of $p = 1.0$.

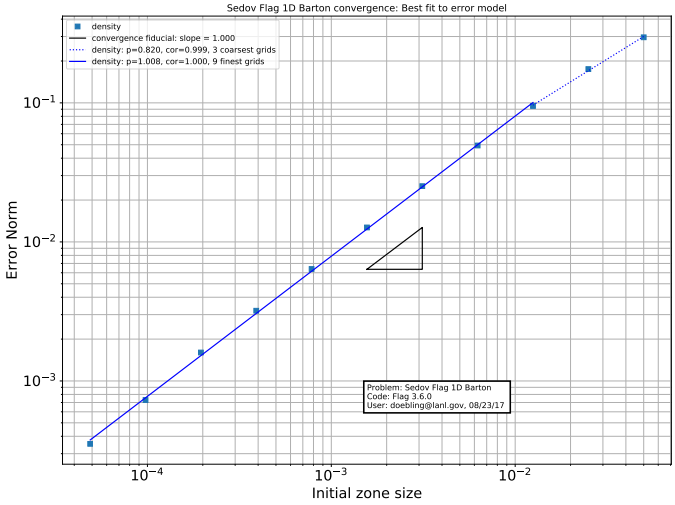


Fig. 11: Convergence of L_1 error norm for Sedov 1D in FLAG, from from $nz = 24$ to $nz = 24576$

- Effect of parameter $massec$:
  The FLAG input parameter $massec$ controls how point masses are computed in FLAG. The default value of $massec = 0$ results in a shock speed that is too fast, resulting in the shock position at $t = 1.0$ that converges to a value about 3% too far from the origin. Using $massec = 2$ causes the shock position to converge to the correct value. The profiles shown in Figure 12 and Figure 13 illustrate this point.

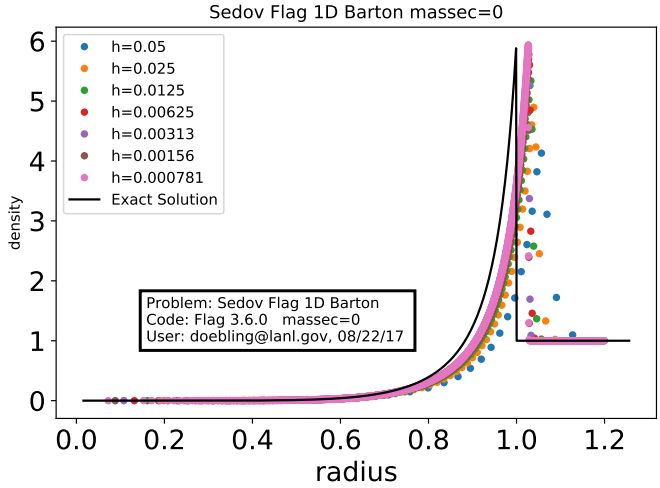


Fig. 12: Sedov in FLAG 1D, with $massec = 0$

- Effect of running a fixed time step
  One question of interest is the sensitivity of spatial convergence results to the use of automatic time step control algorithms. To explore this issue, we ran the Sedov problem in 1D with a maximum time step of 1.0e-6, as compared to the maximum time step under the control of the automatic time step algorithm as shown in Table II (these are determined by the Courant limit). Figure 14 shows that error norms do not change significantly due to the use of the automatic time step controller.

Resolution	dt max
24	1.76e-2
48	7.16e-3
96	2.76e-3
192	1.08e-3
384	4.19e-4
768	1.63e-4
1536	6.38e-5

TABLE II: Values of maximum timestep for each resolution, Sedov 1D in FLAG

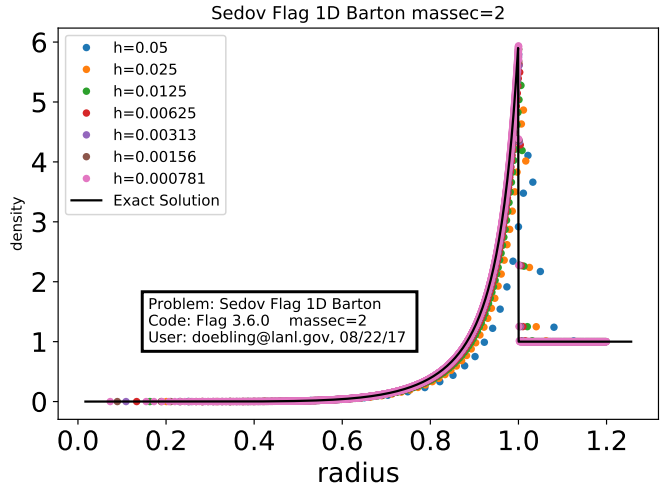


Fig. 13: Sedov in FLAG 1D, with $massec = 2$

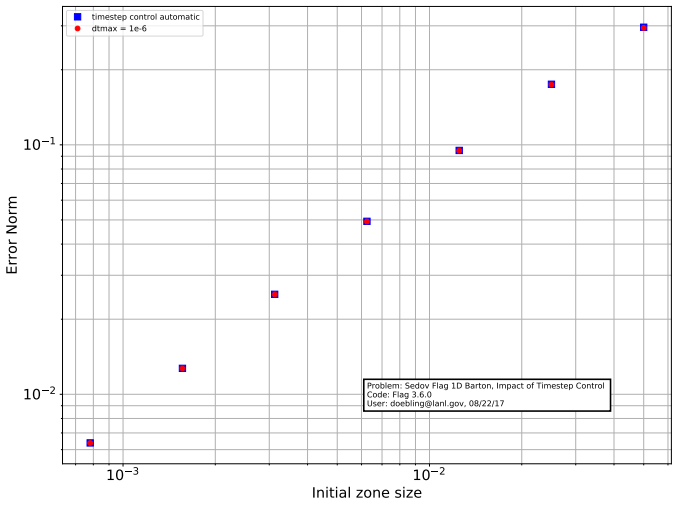


Fig. 14: Impact of controlling timestep for Sedov 1D in FLAG

3) Sedov in FLAG, RZ Cylindrical, Lagrangian:

- Effect of parameter $massec$:
  The default value of $massec = 0$ results in a shock speed that is too fast, resulting in the shock position at $t = 1.0$ that converges to a value about 1% too far from the origin. Using $massec = 2$ causes the shock position to converge to the correct value. The profiles shown in Figure 15 and Figure 16 illustrate this point.
- Choice of q1 and q2 values in Barton model
  Two different sets of choices for q1 and q2 in the Barton artificial viscosity modeled were compared. The first set is $q1 = 1$ and $q2 = 1$. The second set is $q1 = 0$ and $q2 = 2$. The snapshots of density at $t = 1.0$ are shown in Figure 17 and Figure 18. The overshoot in the $q1 = 0$ and $q2 = 2$ case is slightly lower, but both cases suffer from poor symmetry. The error norms for the two cases are shown in Figure 19. The error magnitude in the $q1 = 1$ and $q2 = 1$ case is slightly but consistently smaller. Neither case runs to completion for $nz = 96$ or greater, as the mesh distortion near the axes becomes too great.
- Comparison of results using several artificial viscosity models
  Profiles of the FLAG RZ Sedov density results for various artificial viscosity models are shown in Figure 20. The convergence results for density are shown in Figure 21, pressure in Figure 22, and specific internal energy in Figure 23. From these figures, we can observe significant variation in accuracy and symmetry for the different artificial viscosity models. We also observe consistent convergence for those cases that run at finer grids (Barton and VNR meshes tangle at higher resolutions, i.e. finer than 48 x 48)

4) Sedov in FLAG, 3D Cartesian, Lagrangian: Comparison of results using several artificial viscosity models

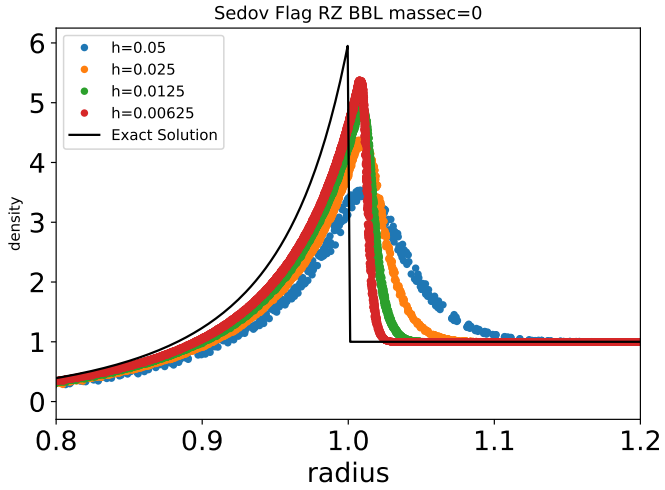


Fig. 15: Sedov FLAG RZ, with $massec = 0$

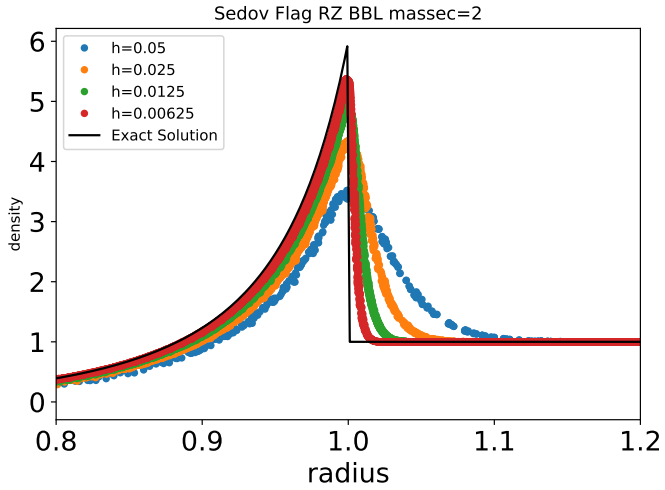


Fig. 16: Sedov in FLAG RZ, with $massec = 2$

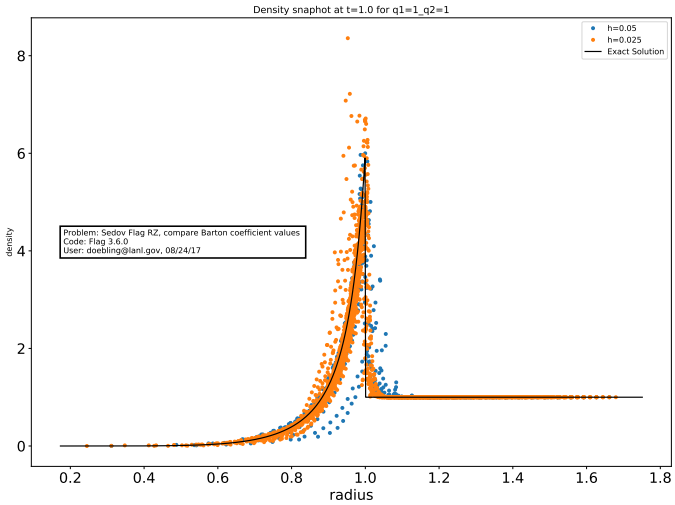


Fig. 17: Density at t=1.0 for Sedov FLAG 1D Barton with $q1 = 1$ and $q2 = 1$

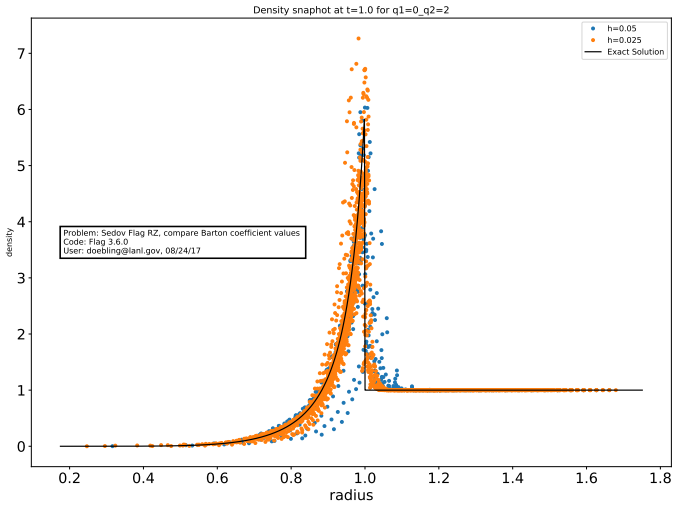


Fig. 18: Density at t=1.0 for Sedov FLAG 1D Barton with $q1 = 0$ and $q2 = 2$

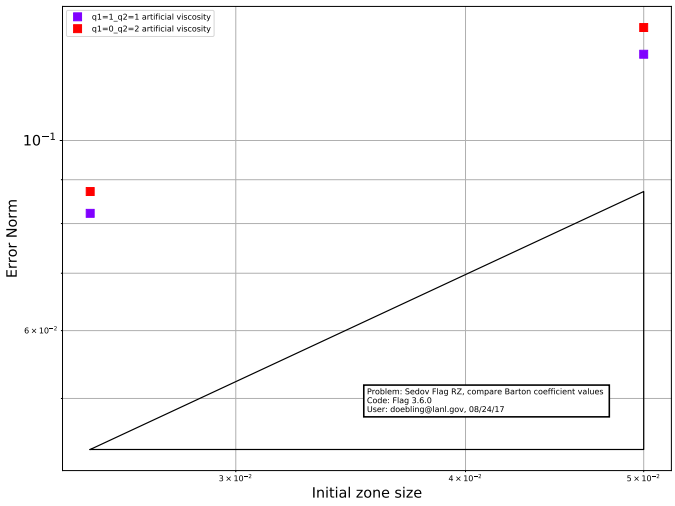


Fig. 19: Error norms of density at t=1.0 for Sedov FLAG 1D Barton with two different $q1$, $q2$ settings

Profiles of the FLAG RZ Sedov density results for various artificial viscosity models are shown in Figure 24. The convergence results for density are shown in Figure 25, pressure in Figure 26, and specific internal energy in Figure 27. From these figures, we observe trends similar to the RZ cases. If anything the 3D cases show more consistent convergence than the RZ cases.

5) Sedov in FLAG, RZ Cylindrical, ALE: In this section, we activate the ALE machinery in FLAG to assess its impact on the Sedov results in RZ cylindrical coordinates. We use the Euler relaxer in default full Eulerian mode, which relaxes the mesh back to its original position after each timestep, thus mimicking the behavior of an Eulerian code. FLAG exhibits significant differences in behavior depending upon our selection of advection options, so this study incorporates four different choices of parameter sets for advection:

1) Code defaults (i.e. no options set, except `errcrit=1.e-5`)
2) Code team recommended settings (`iuse_legacy_ale=0` and `errcrit=1.e-4`)
3) Another set of recommendations from code team, circa 2016. (`errcrit1 = 1.e-4`, `errcrit2 = 1.e-9`, `madvmeth = 31`, `iuse_centroids = 1`, `islopelimiter = 1`, `improved_fct_limiting = 1`, `iaggadvflag = 1`, `iadv_gradient = 0`, `imomadvfix = 1`, `iedmfmeth = 2`, `tole = 1e-20`, `tolf = 1e-20`, `tols = 1e-20`, `betae = 0.1`, `betaf = 0.1`, `betas = 0.1`, `ibrent_adv_fatal = 0`)
4) Settings from #3 plus turn on kinetic energy conservation (`iadvkefixup=1`)

In summary, we observe significant variation in the results depending upon the choice of artificial viscosity model and advection settings. In general, the choice of artificial viscosity may affect symmetry more, while the choice of advection settings may affect the wave arrival time and peak height more, and thus have more impact on the overall accuracy.

It should also be noted that two of the issues observed with pure Lagrangian calculations no longer seem to have much

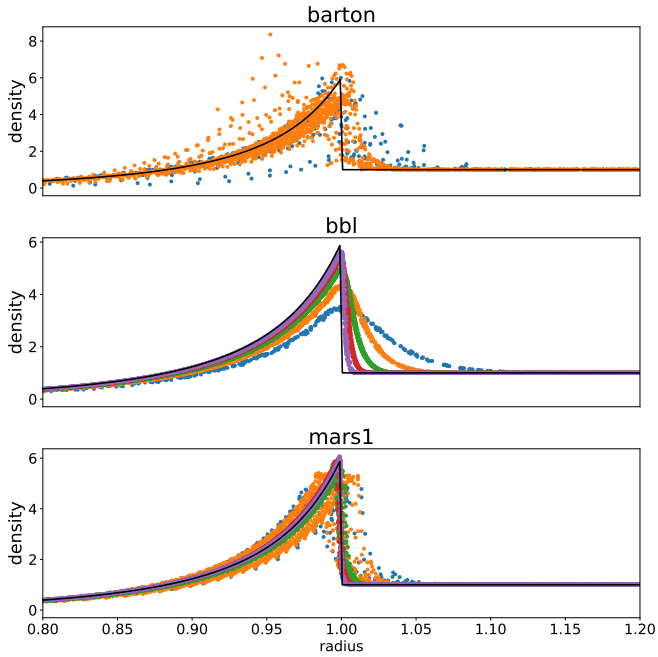


Fig. 20: Snapshots of Density for the Sedov Problem in FLAG RZ for various artificial viscosity models

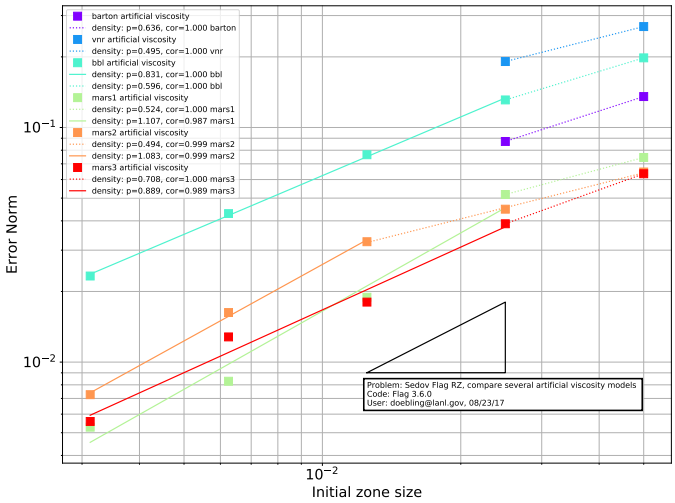


Fig. 21: Convergence of Density for the Sedov Problem in FLAG RZ for various artificial viscosity models

impact when using ALE. First, the parameter `massec` does not have much of an effect, and second, the mesh distortion introduced by the Barton artificial viscosity model doesn't seem to be as much of a factor when using ALE.

One issue identified in the ALE calculations is that the convergence behavior of the algorithm is no longer as well behaved. In fact, some combinations of artificial viscosity model and advection setting do not converge to the correct solution.

A comparison of the results for Barton viscosity over the four sets of ALE settings is shown in Figure 28, with corresponding convergence shown in Figure 29. To illustrate the impact of artificial viscosity model, a comparison of the results for BBL viscosity over the four sets of ALE settings are shown in Figure 30, with corresponding convergence shown in Figure 31. Here, we observe that the fourth set of advection parameters (which includes kinetic energy conservation) clearly shows the best convergence behavior. This begs the question: What does conserving kinetic energy do to the internal energy? The snapshots for specific internal energy are shown in Figure 32, with corresponding convergence shown in Figure 33. Here, we observe that advection set 4 also shows the best accuracy for specific internal energy. So, overall including kinetic energy conservation in the advection seems to produce the best overall result.

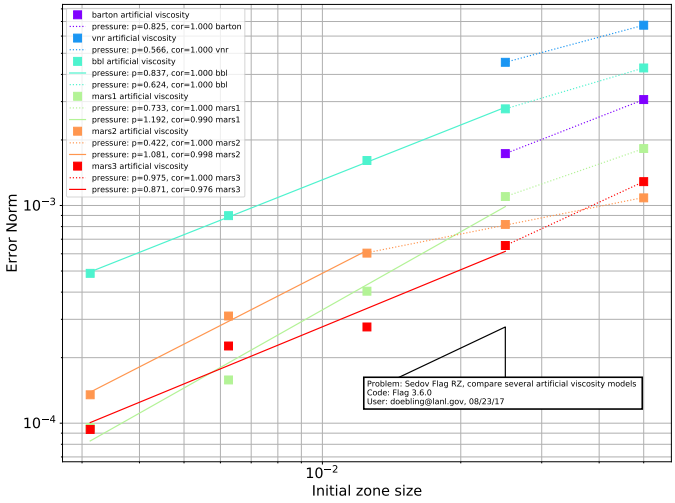


Fig. 22: Convergence of Pressure for the Sedov Problem in FLAG RZ for various artificial viscosity models

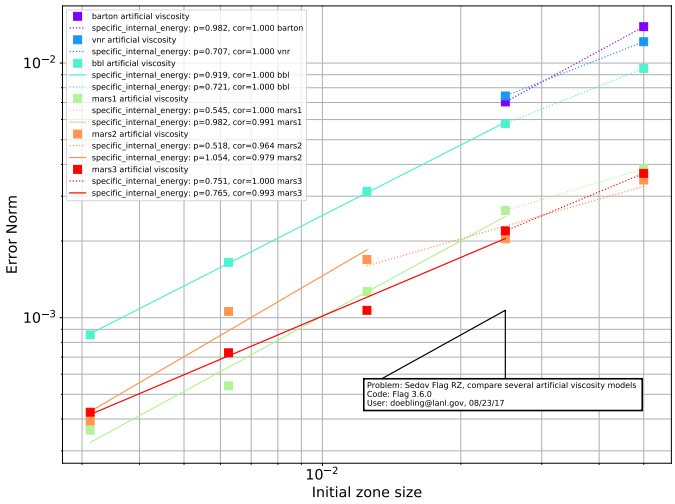


Fig. 23: Convergence of Specific Internal Energy for the Sedov Problem in FLAG RZ for various artificial viscosity models

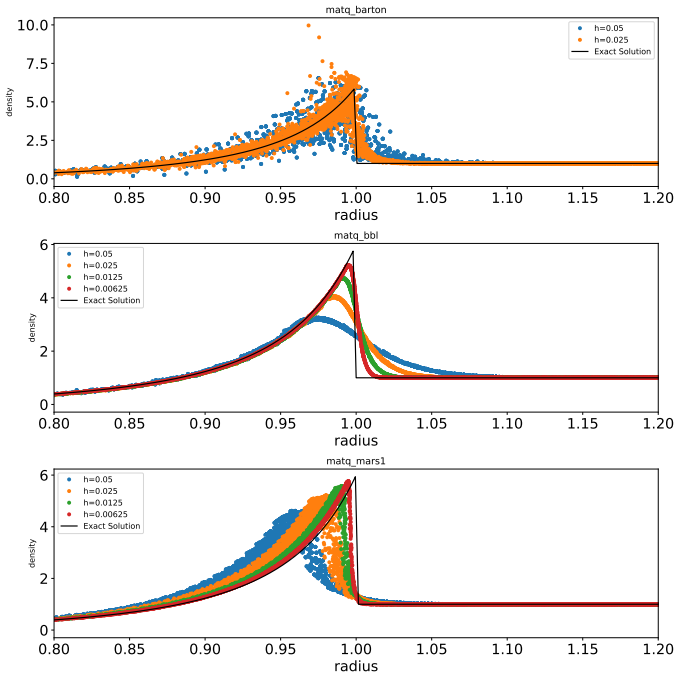


Fig. 24: Snapshots of Density for the Sedov Problem in FLAG 3D for various artificial viscosity models

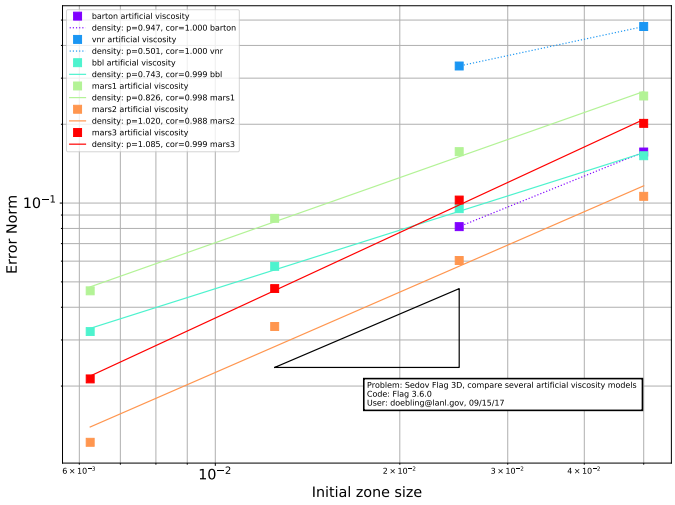


Fig. 25: Convergence of Density for the Sedov Problem in FLAG 3D for various artificial viscosity models

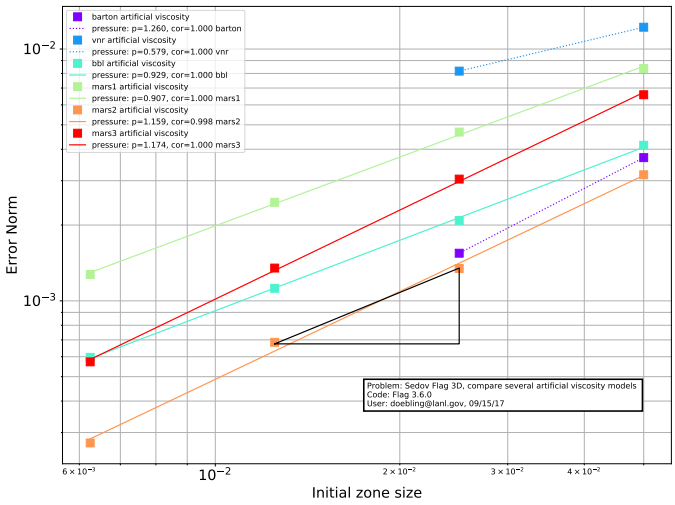


Fig. 26: Convergence of Pressure for the Sedov Problem in FLAG 3D for various artificial viscosity models

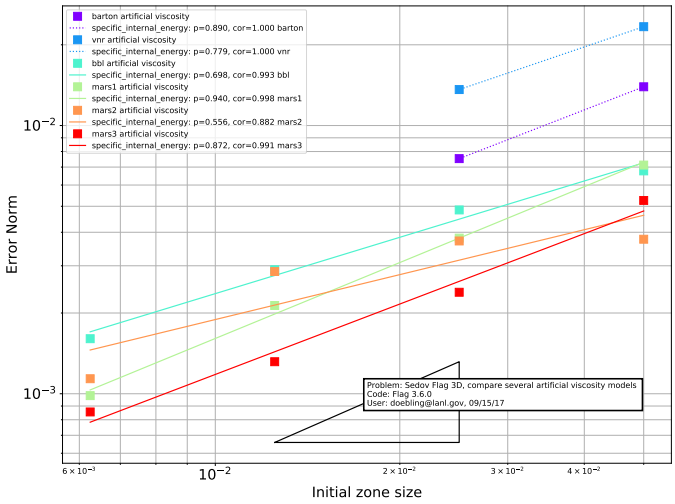


Fig. 27: Convergence of Specific Internal Energy for the Sedov Problem in FLAG 3D for various artificial viscosity models

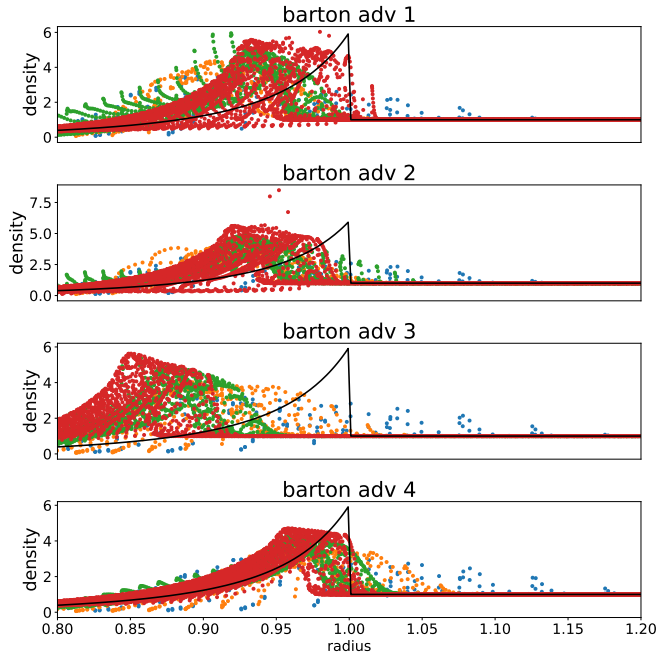


Fig. 28: Snapshots of density for the Sedov Problem in FLAG RZ using Barton viscosity and the Euler relaxer with various advection settings

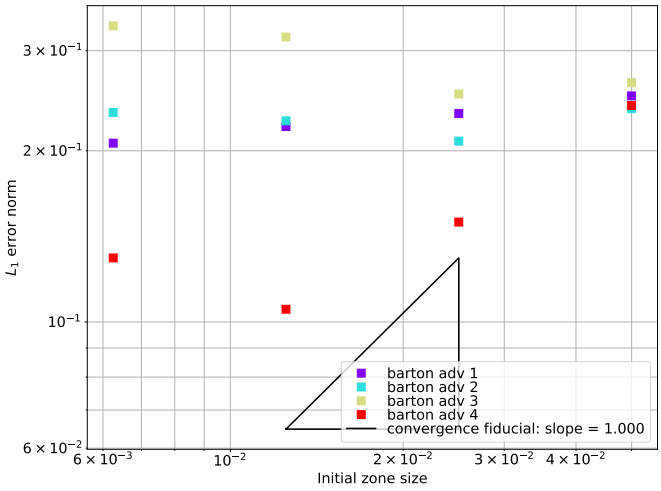


Fig. 29: Convergence of density for the Sedov Problem in FLAG RZ using Barton viscosity and the Euler relaxer with various advection settings

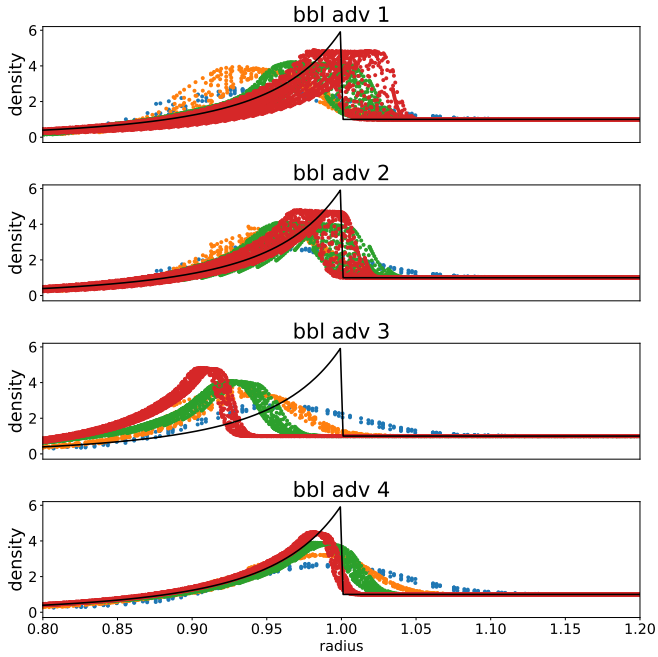


Fig. 30: Snapshots of density for the Sedov Problem in FLAG RZ using BBL viscosity and the Euler relaxer with various advection settings

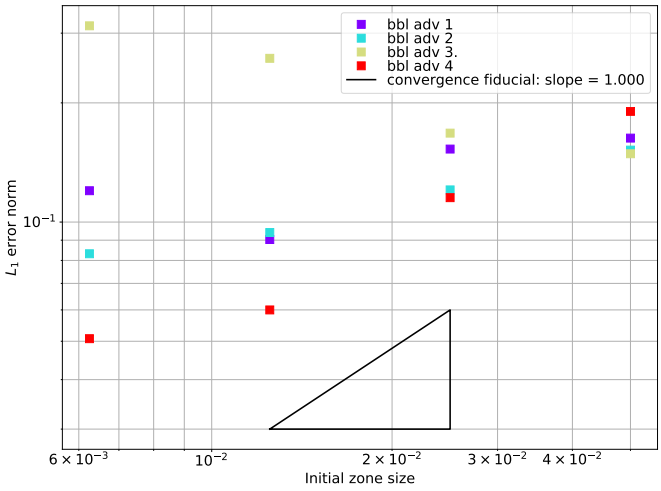


Fig. 31: Convergence of density for the Sedov Problem in FLAG RZ using BBL viscosity and the Euler relaxer with various advection settings

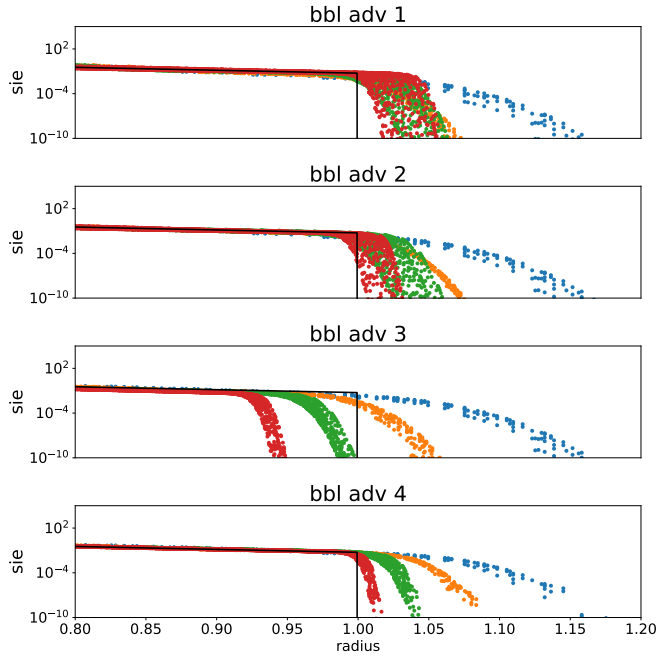


Fig. 32: Snapshots of Specific Internal Energy for the Sedov Problem in FLAG RZ using the Euler relaxer with various advection settings

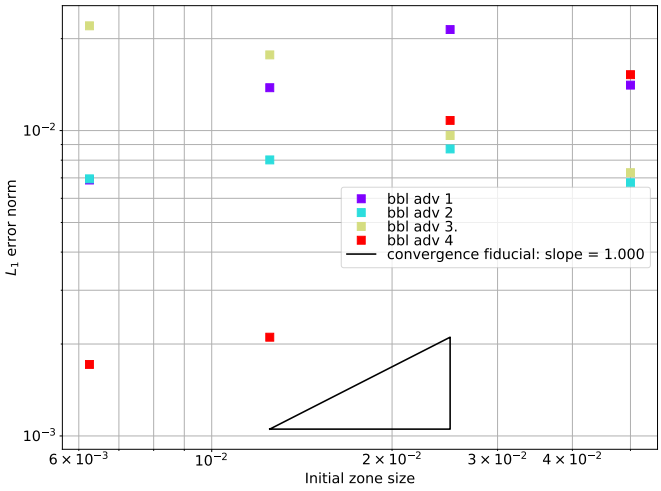


Fig. 33: Convergence of Specific Internal Energy for the Sedov Problem in FLAG RZ using the Euler relaxer with various advection settings

D. FLAG Riemann Shock Tube Problems

1) Summary of Observations: Six shock tube problems are analyzed in FLAG, pure Lagrange, on a 1D Cartesian mesh. Overall, FLAG performs better than expected. Tests 1, 2, 3, 5, and 6 all converge at first order. Test 4 (The Slow-Moving Shock problem) converges at a slower rate and also drives the code to a much smaller timestep than the others do.

2) Sod Shock Tube Problem (Riemann Test 1) in FLAG, 1D Cartesian: The Sod shock tube problem is well known and involves two ideal gases of different initial states side-by-side. Running Riemann Test 1 with 11 levels of resolution (from $nz = 50$ to $nz = 51200$) yields the results shown in Figure 34. The error norm is clearly in the asymptotic convergence region for all resolutions tested, and converges at a rate of $p = 0.985$, which we interpret as first order. While we would expect a rate of $p = 2/3$ on this problem that has a contact discontinuity, we attribute the higher convergence rate to the fact that this is a Lagrangian calculation, and the convergence theories assume a fixed grid. For a Lagrangian calculation, the discontinuity is always at a cell boundary.

3) Einfeldt (or 1-2-3) Problem (Riemann Test 2) in FLAG, 1D Cartesian: The Einfeldt problem involves two fluids moving away from each other. It develops a double rarefaction where the area in the center approaches a vacuum over time. The rarefaction is C_1 continuous because there is an undefined second derivative at the head of the rarefaction. Thus, we expected first order convergence. Running Riemann Test 2 with 11 levels of resolution (from $nz = 50$ to $nz = 51200$) yields the results shown in Figure 35. The error norm is clearly in the asymptotic convergence region for all resolutions tested, and converges at a rate of $p = 0.948$, which we interpret as first order.

4) Stationary Contact Problem (Riemann Test 3) in FLAG, 1D Cartesian: The stationary contact problem consists of a strong shock wave moving to the right, a stationary contact, and a strong rarefaction moving to the left. As with the Sod shock tube, we would normally expect $p = 2/3$ because of the contact discontinuity. Running Riemann Test 3 with 11 levels of resolution (from $nz = 50$ to $nz = 51200$) yields the results shown in Figure 36. The error norm is clearly in the asymptotic convergence region for all resolutions tested, and converges at a rate of $p = 0.989$, which we interpret as first order.

5) Slow Shock Problem (Riemann Test 4) in FLAG, 1D Cartesian: The slow shock problem consists of a Mach 3 shock wave moving slowly to the right; some numerical methods exhibit unphysical oscillations behind the shock. As with the Sod shock tube, we would normally expect $p = 2/3$ because of the contact discontinuity. Running Riemann Test 4 with 11 levels of resolution (from $nz = 50$ to $nz = 51200$) yields the results shown in Figure 37. Unlike the other five Riemann tests, the error norm starts outside of the asymptotic range, then appears to go asymptotic to $p = 0.90$ at $nz = 400$, only to change convergence rates again at $nz = 6400$ to $p = 0.456$. Examining the profiles for this case, there are some unphysical oscillations visible around -1.5 and also around -0.5. Further investigation of this test problem is warranted.

6) Shock-Contact-Shock Problem (Riemann Test 5) in FLAG, 1D Cartesian: The shock-contact-shock problem consists of two shocks separating from each other. It is similar to a planar version of the Noh problem, but with weaker shocks. As with the Sod shock tube, we would normally expect $p = 2/3$ because of the contact discontinuity. But since the contact discontinuity is stationary, we might expect the shock problem rate of $p = 1$ to dominate. Running Riemann Test 5 with 11 levels of resolution (from $nz = 50$ to $nz = 51200$) yields the results shown in Figure 38. The error norm is clearly in the asymptotic convergence region starting at $nz = 200$, and converges at a rate of $p = 0.976$, which we interpret as first order.

7) LeBlanc Problem (Riemann Test 6) in FLAG, 1D Cartesian: The Leblanc problem is identical to the Sod problem, with much larger pressure and energy differentials, that result in a stronger shock and rarefaction. As with the Sod shock tube, we would normally expect $p = 2/3$ because of the contact discontinuity. Running Riemann Test 6 with 11 levels of resolution (from $nz = 50$ to $nz = 51200$) yields the results shown in Figure 39. The error norm is clearly in the asymptotic convergence region starting at $nz = 100$, and converges at a rate of $p = 1.004$, which we interpret as first order.

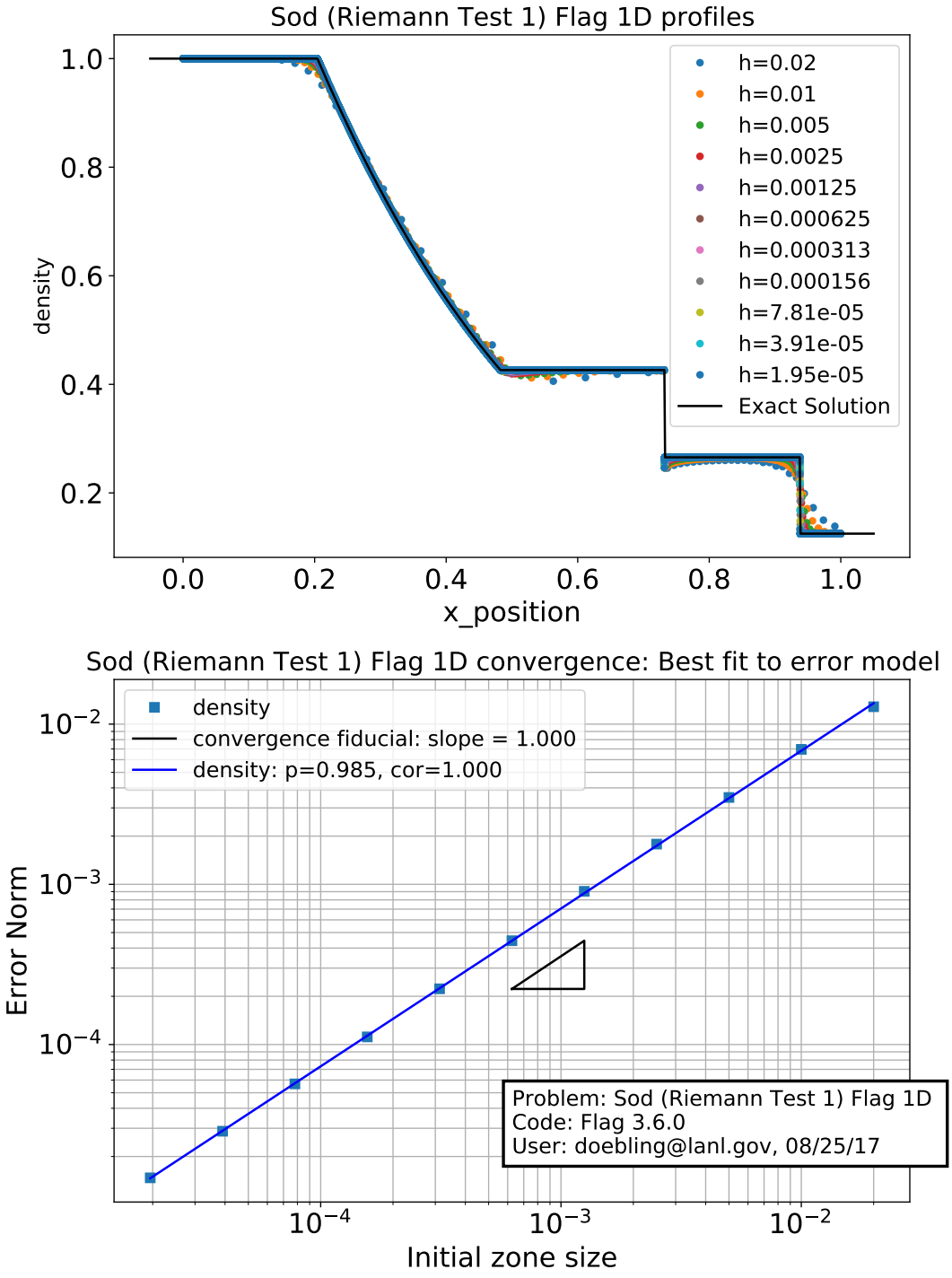


Fig. 34: Convergence of L_1 error norm for Sod Shock Tube (Riemann Test 1) in FLAG, from from $nz = 24$ to $nz = 51200$

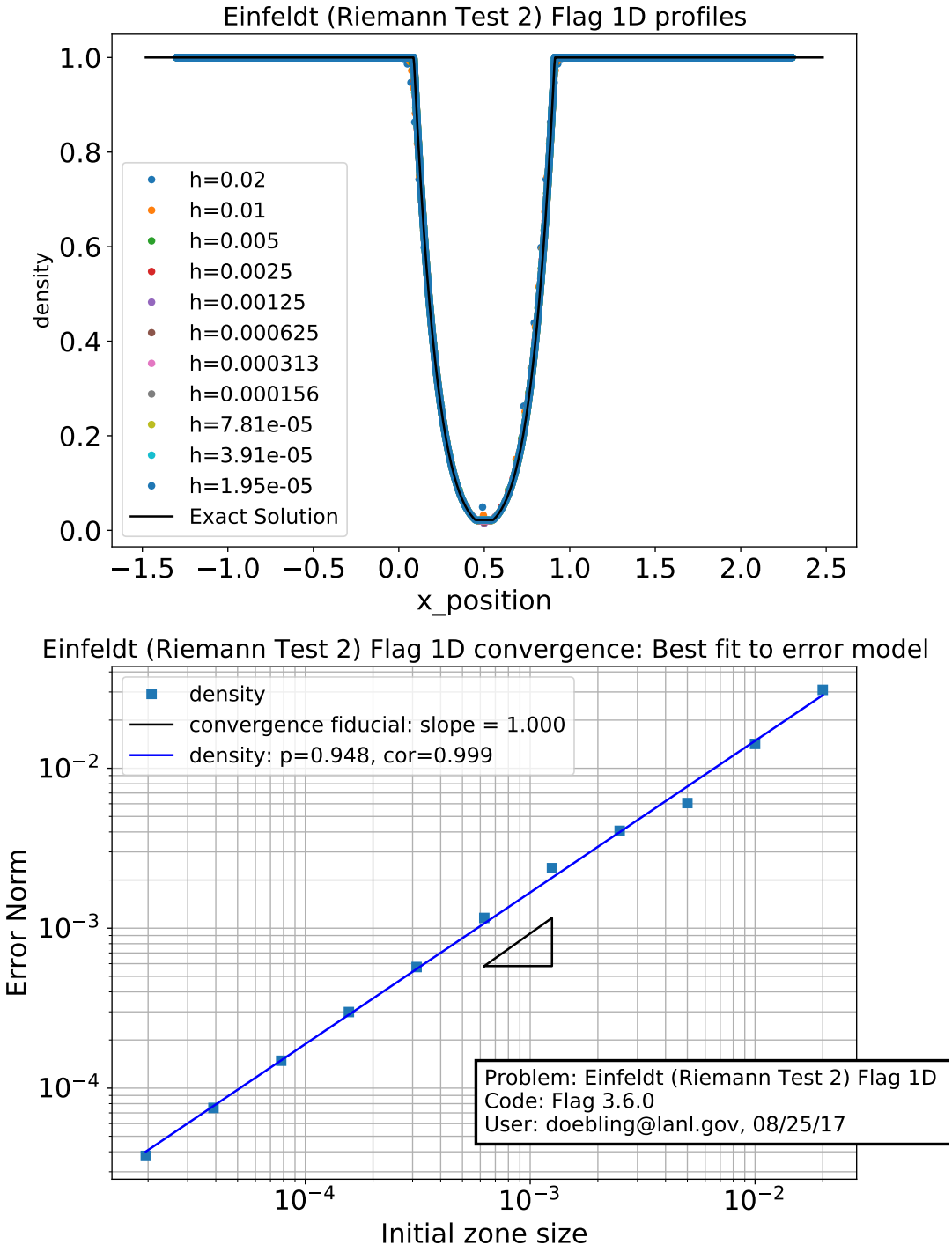


Fig. 35: Convergence of L_1 error norm for the Einfeldt Problem (Riemann Test 2) in FLAG, from from $nz = 24$ to $nz = 51200$

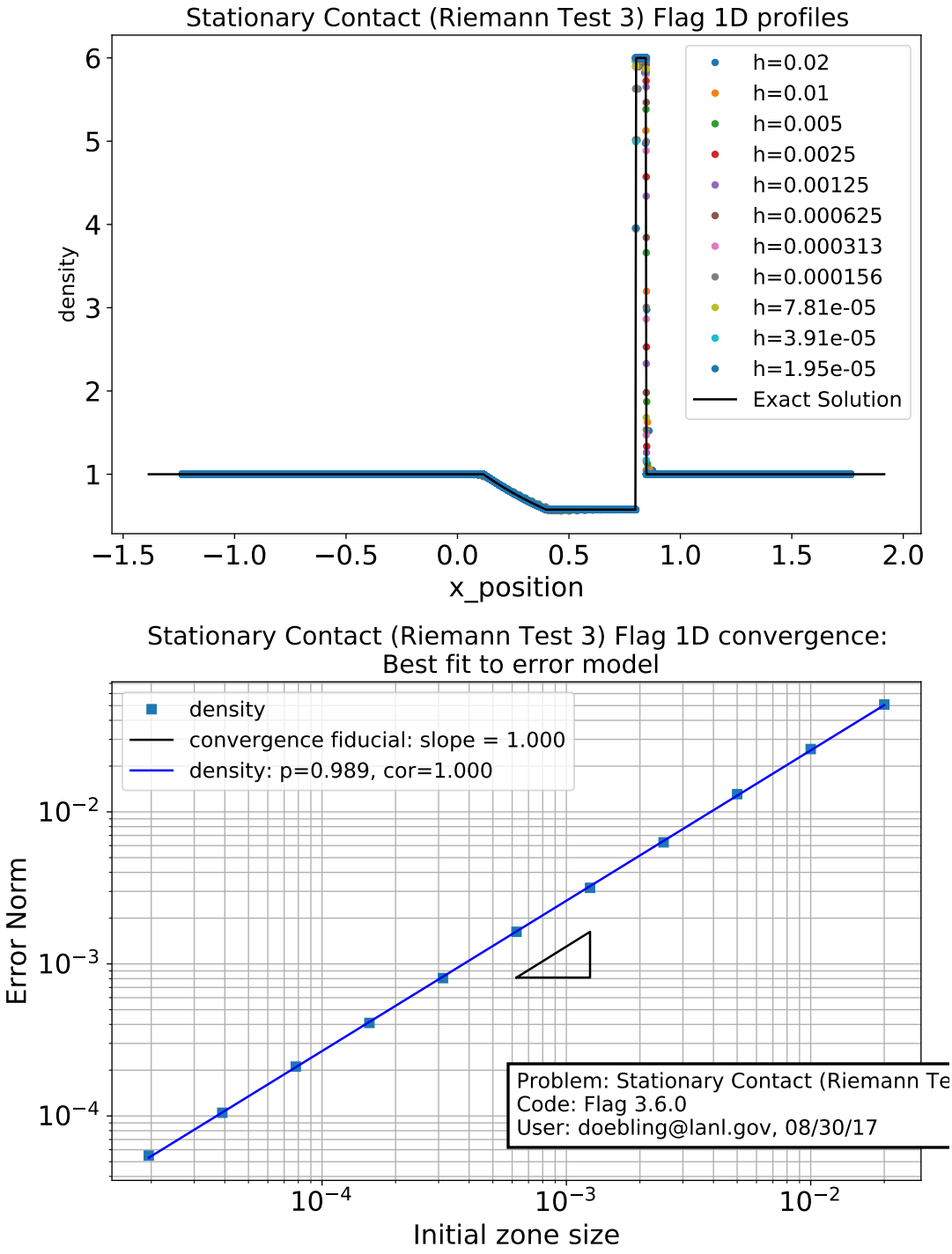


Fig. 36: Convergence of L_1 error norm for the Stationary Contact Problem (Riemann Test 3) in FLAG, from from $nz = 24$ to $nz = 51200$

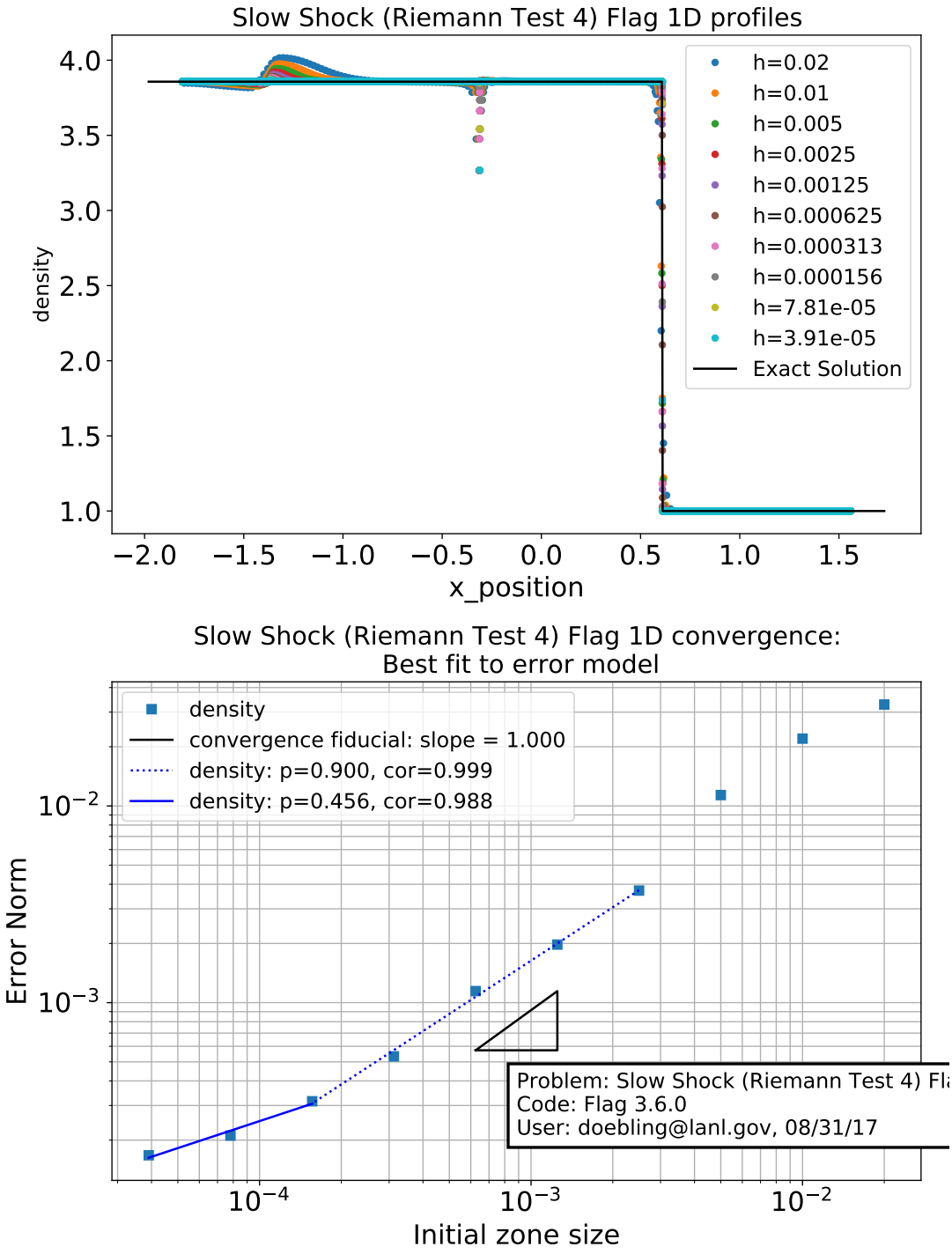


Fig. 37: Convergence of L_1 error norm for the Slow Shock Problem (Riemann Test 4) in FLAG, from from $nz = 24$ to $nz = 51200$

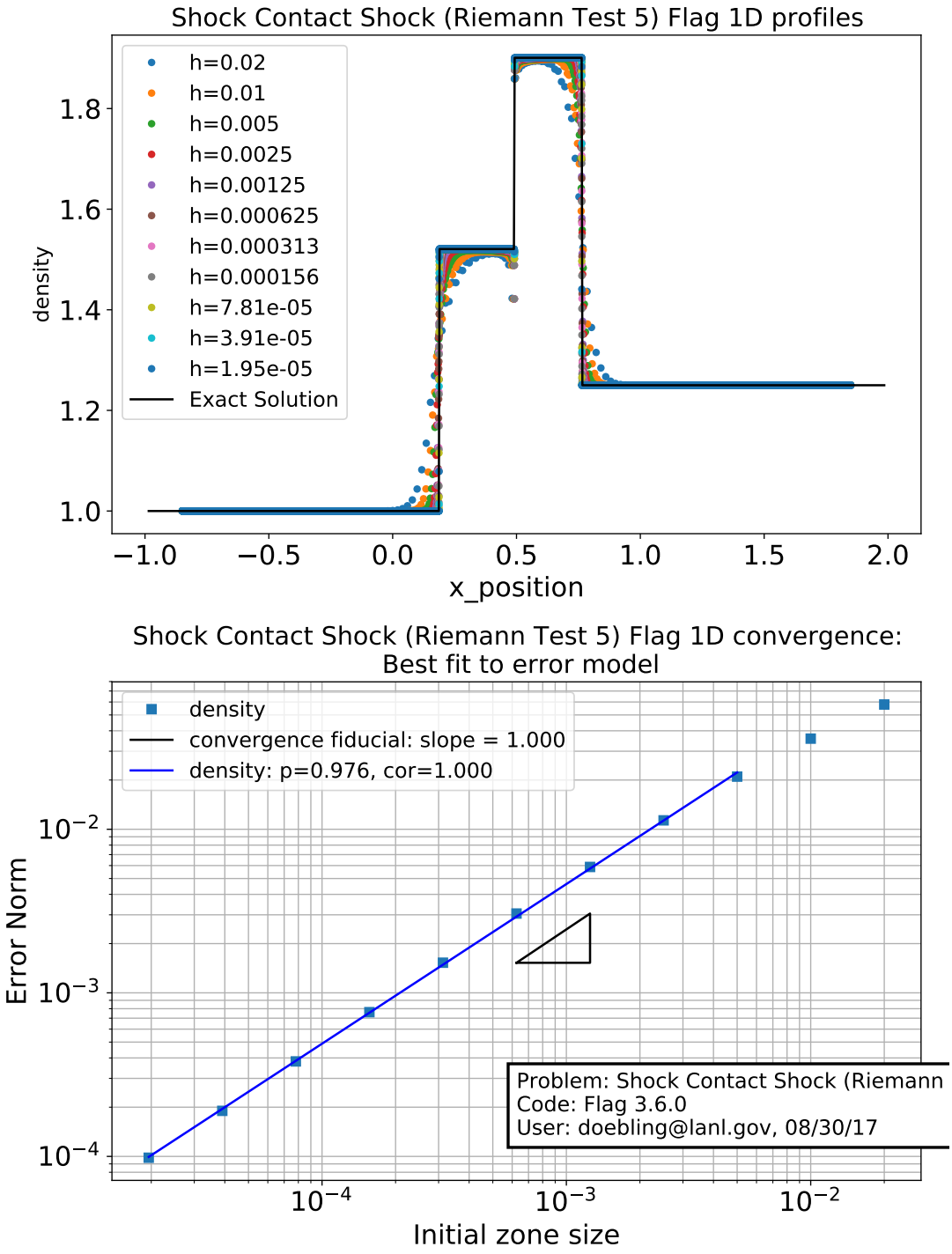


Fig. 38: Convergence of L_1 error norm for the Shock Contact Shock Problem (Riemann Test 5) in FLAG, from from $nz = 24$ to $nz = 51200$

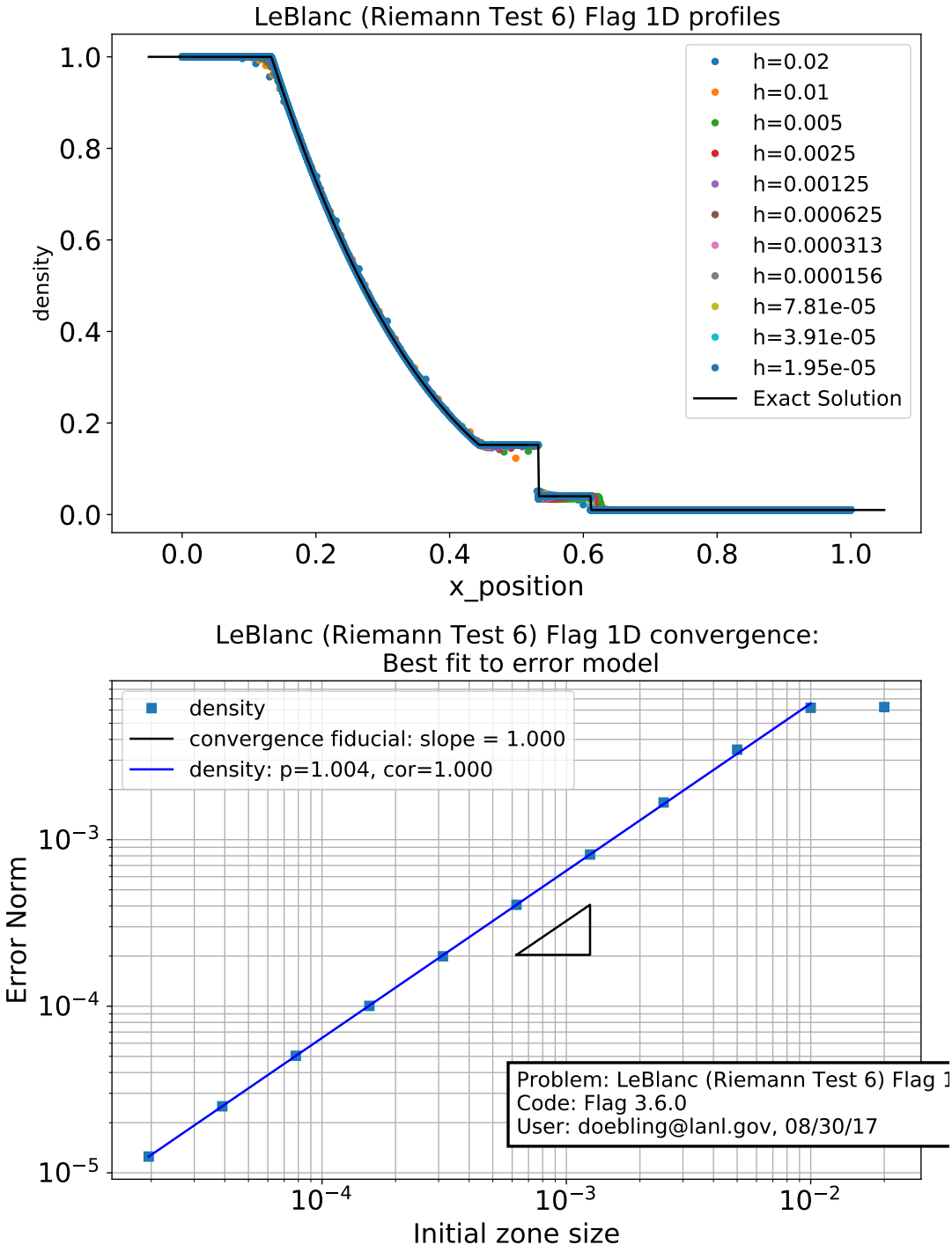


Fig. 39: Convergence of L_1 error norm for the Leblanc problem (Riemann Test 6) in FLAG, from from $nz = 24$ to $nz = 51200$

E. FLAG Noh Problem

In FLAG, the Noh problem grids are set up as specified in the project standard definition [4].

1) Summary of Observations:

- Noh is a pathological test case for edge-based viscosity like Barton
- VNR converges consistently for sufficient resolution
- BBL and MARS methods converge consistently at all resolutions

2) Noh in FLAG, RZ Cylindrical, Lagrangian: The density for Noh in RZ cylindrical coordinates for various artificial viscosity models is shown in Figure 40 (snapshots) and Figure 41 (convergence). Here we observe that the BBL and MARS methods all convergence in the asymptotic regime right from the coarsest resolution. Additionally, VNR convergences after resolution of $nz = 96$ is achieved, at an apparent rate greater than $p = 1$. It should be noted that the Barton method does not appear on this plots past the coarsest resolution. The Noh problem in RZ and 3D appears to be an almost pathological case for edge-based viscosity methods such as Barton, as all finer grids exhibit mesh tangling.

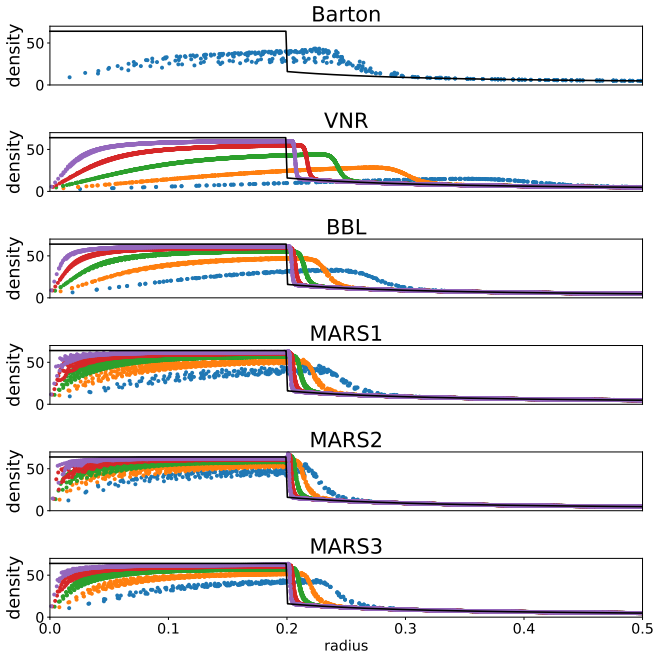


Fig. 40: Snapshots of Density for the Noh Problem in FLAG RZ for various artificial viscosity models

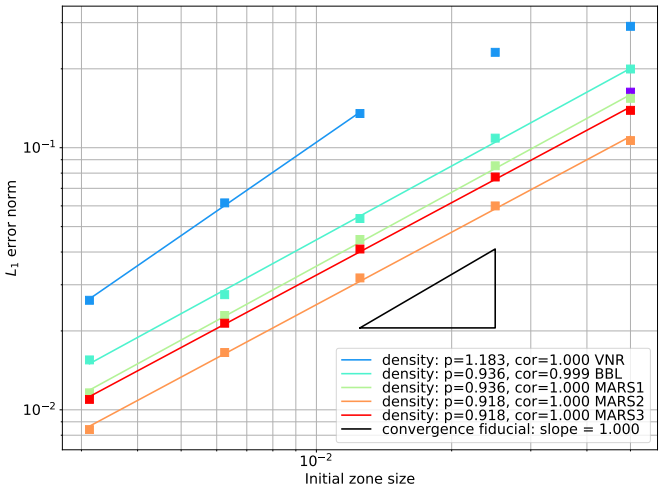


Fig. 41: Convergence of Density for the Noh Problem in FLAG RZ for various artificial viscosity models

3) Noh in FLAG, 3D Cylindrical, Lagrangian: The density for Noh in 3D cylindrical coordinates for various artificial viscosity models is shown in Figure 42 (snapshots) and Figure 43 (convergence). Here we observe, similar to the RZ results, VNR converges as expected once we reach a resolution of $nz = 48$. BBL, MARS1, and MARS3 all converge at the expected rate and, as with RZ, are in the asymptotic regime right from the coarsest resolution. MARS2 is accurate as well, but does not converge at the expected rate in 3D.

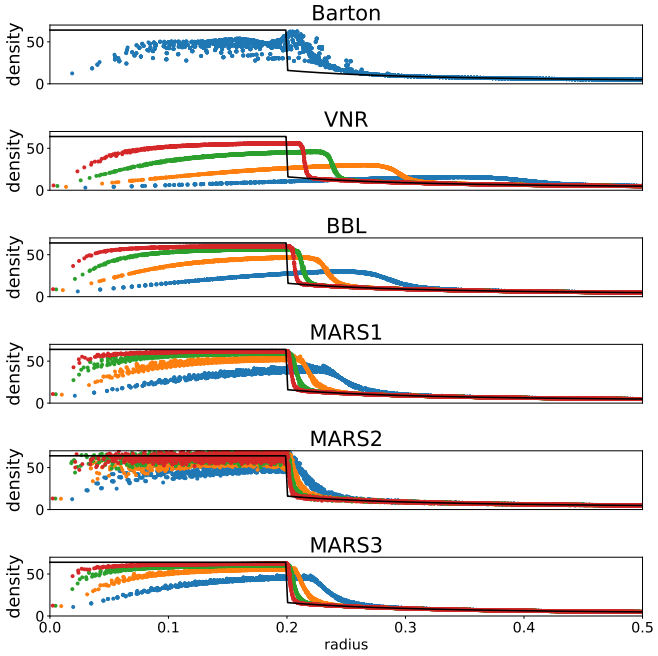


Fig. 42: Snapshots of Density for the Noh Problem in FLAG 3D for various artificial viscosity models

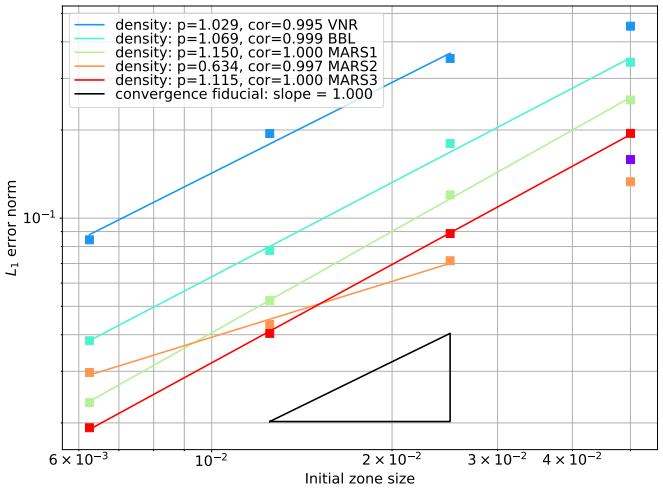


Fig. 43: Convergence of Density for the Noh Problem in FLAG 3D for various artificial viscosity models

F. xRAGE Hydrodynamics

This section presents the verification assessment of the LANL multi-physics hydro code xRAGE for three primary hydro schemes: (i) Lagrange + Remap (LR), (ii) Direct Eulerian (DE), and (iii) an unspilt method that is currently under development (referred to as UnSplit in this document). For LR and DE, we exercise the code options of pressure-temperature equilibrium (PTE) and non-pressure-temperature equilibrium (noPTE). This report will mostly emphasize PTE with LR and DE. For the unspilt option, one can choose between two Riemann solvers, HLL and HLLC. The various xRAGE options are listed in Table III, and give the following six hydro variants, with the default combination being LR/PTE.

TABLE III: Hydro options for xRAGE. The xRAGE default combination is LR/PTE.

option	**LR/PTE**	DE/PTE	LR/noPTE	DE/noPTE	UnSplit/HLL	UnSplit/HLLC
hydro_version	2	2	14	14	2016	2016
hydro_option_lr	1	0	1	0	-	-
unsplit_riemann	-	-	-	-	2	3
unsplit_integrator	-	-	-	-	2	2

We have performed convergence analyses for all 6 combinations, in both 1D-spherical and 2D-RZ geometries, giving 12 possible option settings per problem type. Due to the large number of cases, we shall concentrate primarily on DE and LR with PTE and UnSplit with HLLC, although representative cases for UnSplit/HLL will also be presented in 2D-RZ geometry. The results for LR/DE with noPTE are identical to those of PTE, and will be not be presented presented here. This is because the test problems can be expressed in terms of a single-material model. We have, however, performed the corresponding analyses using multiple material input decks, and we find identical results compared with the single material decks (although the run times are significantly longer for the noPTE option).

G. xRAGE Sedov Problem

1) Summary of Observations: Results for the 1D Sedov runs are shown in Figs. (44)-(49) for the options LR/PTE, DE/PTE, and UnSplit/HLLC. These studies employ 8 levels of spatial resolution refinement, with $N = 24, 48, 96, 192, 384, 768, 1536, 3072$. Figures (44), (46), and (48) overlay the numerical solutions on top of the exact solution for the density, velocity, and pressure profiles, while Figs. (45), (47), and (49) provide the corresponding convergence analyses. Note that, at lower resolutions $N = 24 - 192$, the numerical solutions start in a non-asymptotic regime with a convergence rate of $p \approx 0.5$. However, at the $5^{\rm th}$ refinement, $n = 384$, the solutions turn over and enter the asymptotic regime, with convergence rates $p = 0.917, 0.911,$ and 0.963 (for LR, DE, and UnSplit), which we interpret as $p = 1$. Note that the UnSplit hydro (the red line in the figures) is a factor of 2 more accurate than the split hydro schemes of LR and DE. In every test problem that we have considered, the UnSplit hydro is generally a factor of 2 better than the split hydro.

Results for the 2D Sedov runs are presented in Figs. (50)-(55), and they show qualitatively similar behavior as the 1D results. As before, we exercise the options LR/PTE and DE/PTE. We augment the 1D analysis by showing both HLL and HLLC in the UnSplit case (HLL and HLLC are identical in 1D). The UnSplit hydro is again twice as accurate as LR/DE, for both HLL and HLLC. In Figs. 56 and 57 we provide density contour plots for the LR and LE method, along with line-outs at selected angles. The methods are quite symmetric. Similarly, Fig. 57 shows the contour plots for 2D-RZ geometry with the UnSplit hydro, for the options HLL and HLLC. Note the "carbuncle" instability in the HLLC method (right panel, upper y-axis and far right x-axis). This is a well known and purely numerical instability that occurs when a shock aligns with the numerical grid (in this case, along the x- and y-axes). This instability can be removed by adding some dissipation along the aligned grids, an approach which is currently being implemented in xRAGE. Finally, note that Fig. (52) shows considerable asymmetry at small values of the velocity in 2D, although this is not observable in the convergence analysis of Fig. (53). This asymmetry has a negligible effect on the hydrodynamics since the corresponding momentum is small.

2) Sedov in xRAGE, 1D Spherical: Figures (44)-(49) illustrate the xRAGE solutions of the 1D spherical Sedov problem for the density, velocity, and pressure respectively, and their corresponding convergence analyses.

3) Sedov in xRAGE, 2D-RZ : Figures (50)-(55) illustrate the solutions and their corresponding convergence analyses for the density, velocity, and pressure in 2D-RZ geometry for the spherical Sedov problem.

Figure (52) shows that there is considerable asymmetry at small values of velocity in 2D-RZ geometry, although this is not observable in the convergence analysis of Fig. (53). These values are small very small, however, and cary negligible momentum.

Figure 56 shows the contour plots for 2D-RZ geometry with the LR and DE options, and their corresponding line-out graphs at selected angles in the RZ-plane. The solutions shown in the Figure corresponds to $N = 192$.

Figure 57 shows the contour plots for 2D-RZ geometry with the UnSplit hydro, for the options HLL and HLLC. The solution shown in the Figure corresponds to $N = 192$. The HLLC (right panel) shows the "carbuncle" instability, a well known and purely numerical instability that occurs when a shock is aligned with the numerical grid. This instability can be removed by adding a small amount of dissipation along the aligned grids.

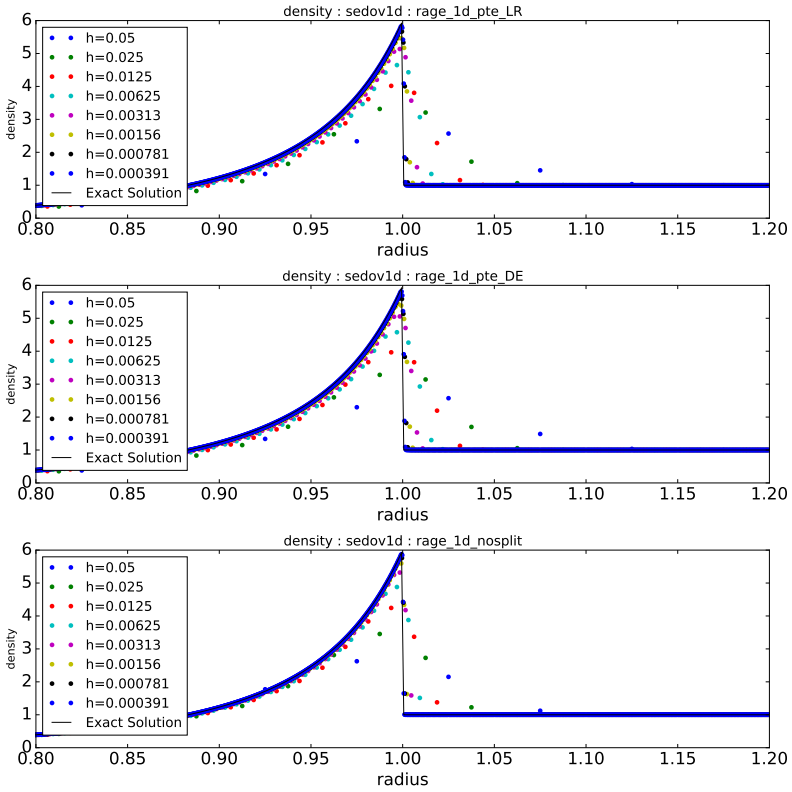


Fig. 44: Density for 1D Sedov with hydro schemes LR, DE, and UnSplit.

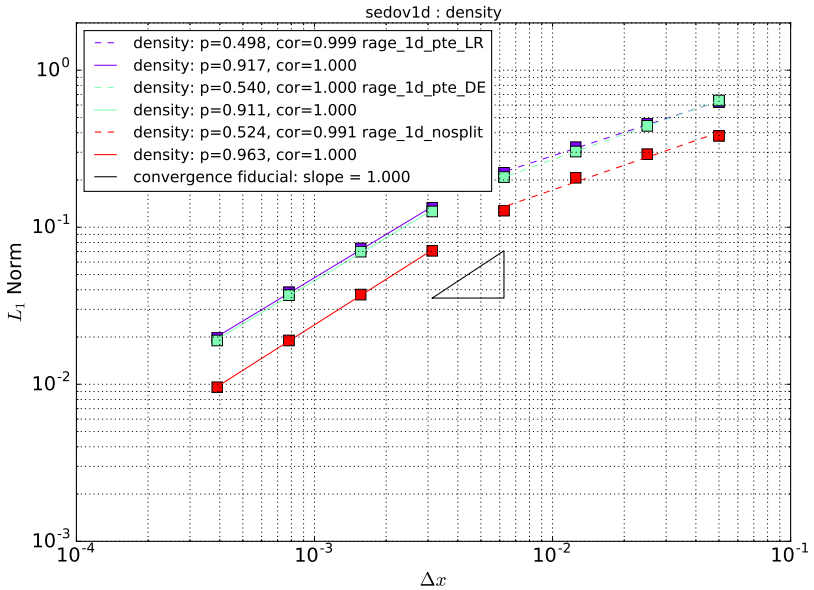


Fig. 45: Corresponding convergence analysis for density with hydro schemes LR, DE, and UnSplit.

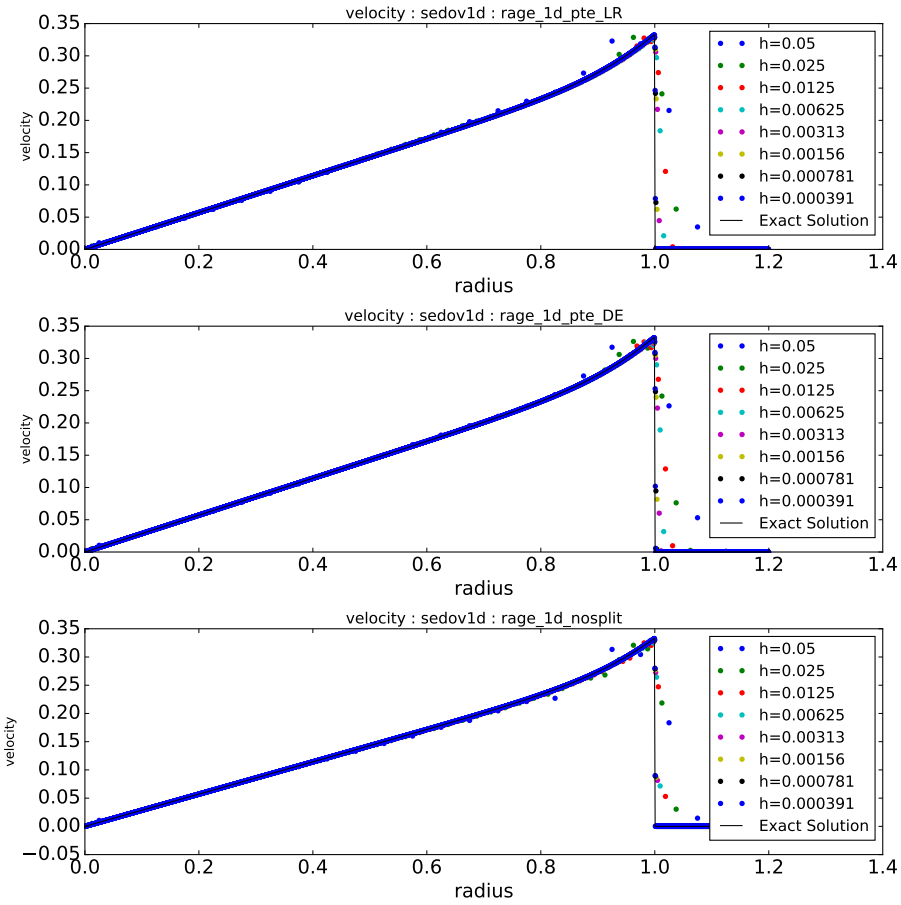


Fig. 46: Velocity for 1D Sedov with hydro schemes LR, DE, and UnSplit.

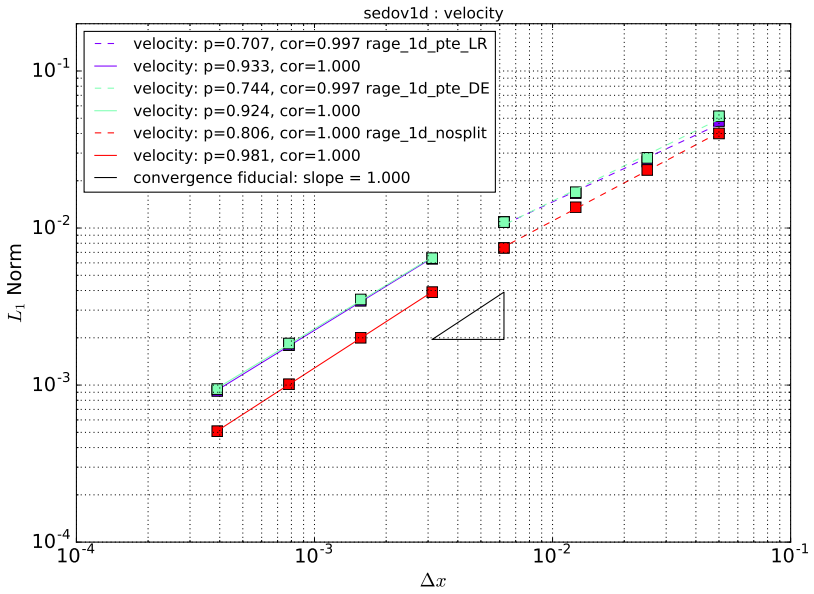


Fig. 47: Convergence analysis for velocity with hydro schemes LR, DE, and UnSplit.

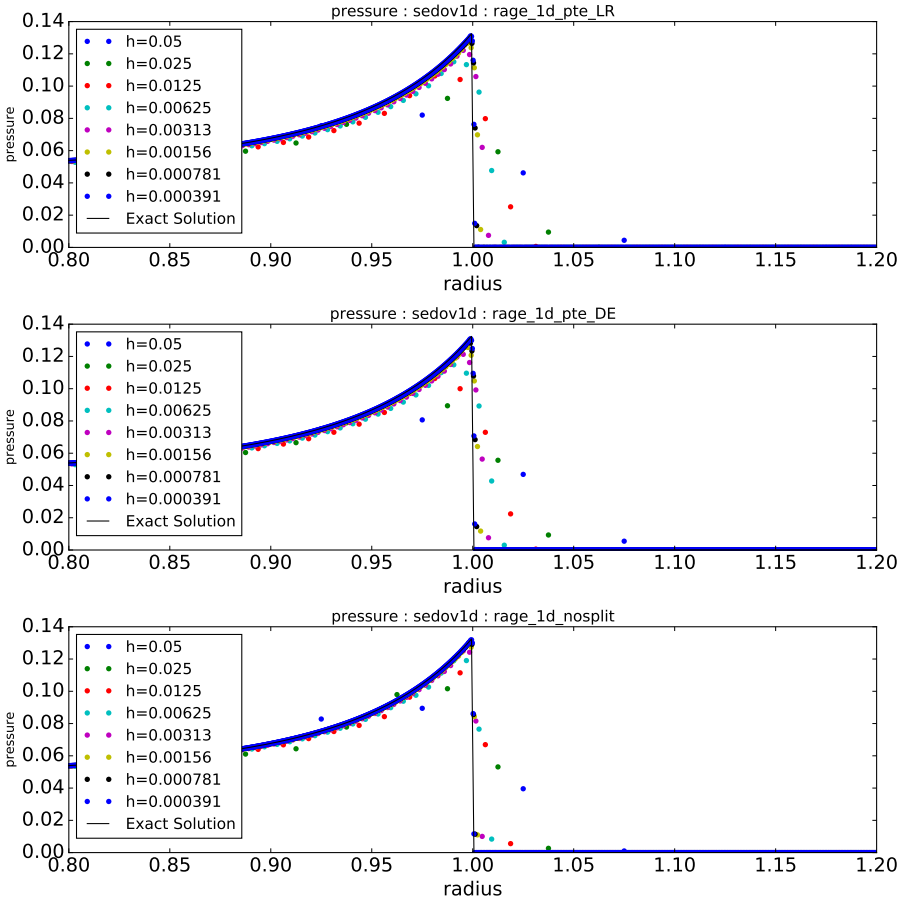


Fig. 48: Pressure for 1D Sedov with hydro schemes LR, DE, and UnSplit.

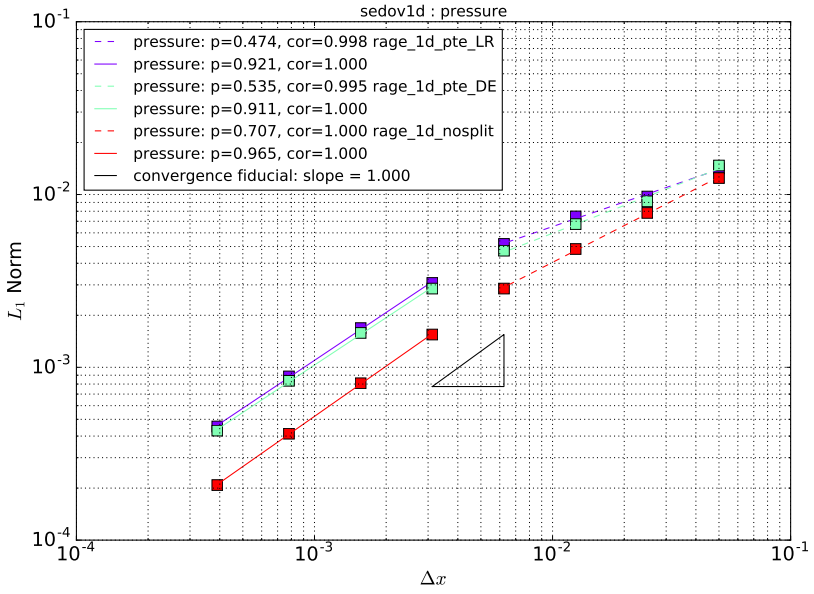


Fig. 49: Convergence analysis for pressure with hydro schemes LR, DE, and UnSplit.

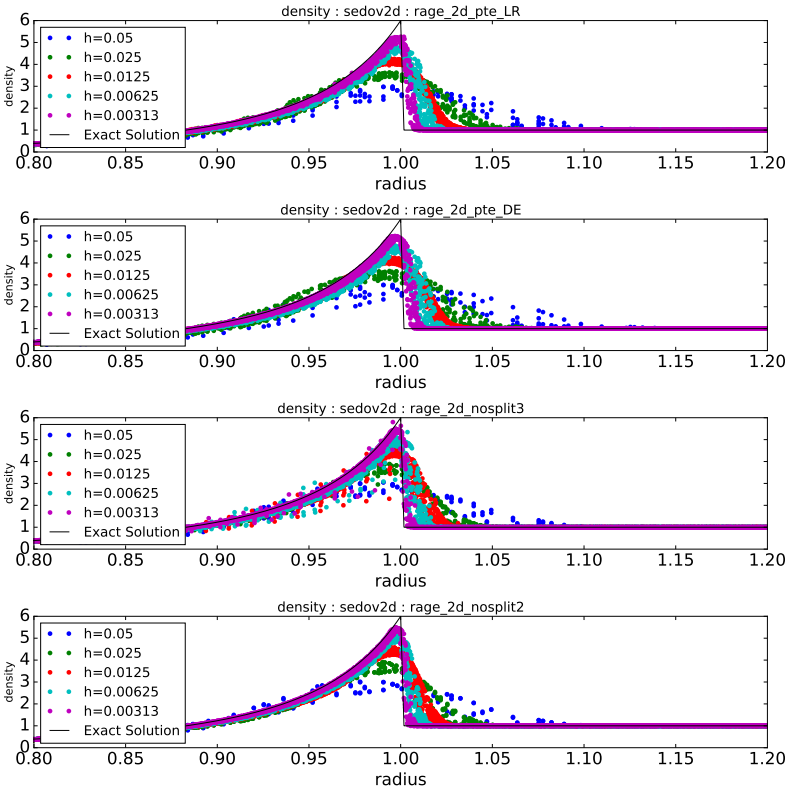


Fig. 50: Density for 2D-RZ Sedov with hydro schemes LR, DE, and UnSplit.

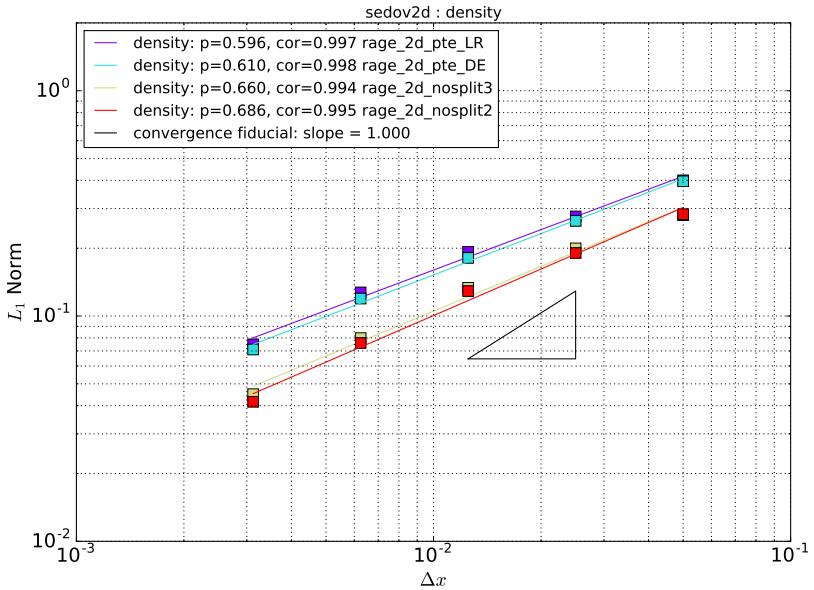


Fig. 51: Convergence analysis for density for 2D Sedov for the LR, DE, and UnSplit hydro.

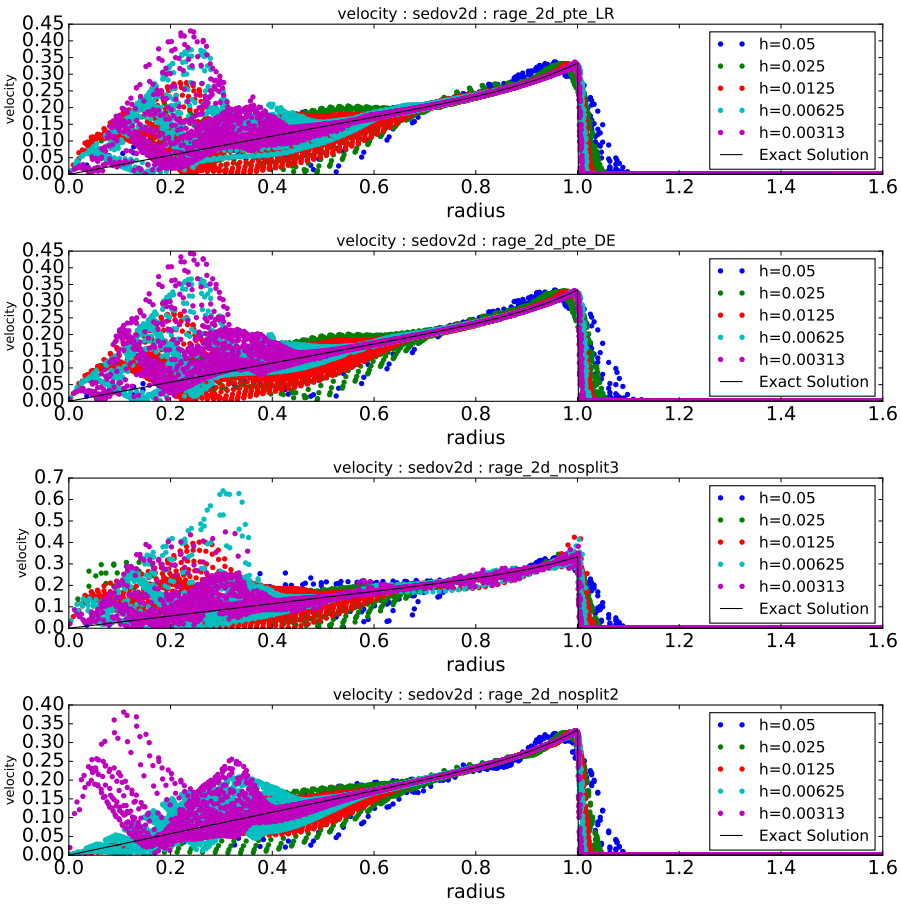


Fig. 52: Velocity for 2D-RZ Sedov with hydro schemes LR, DE, and UnSplit.

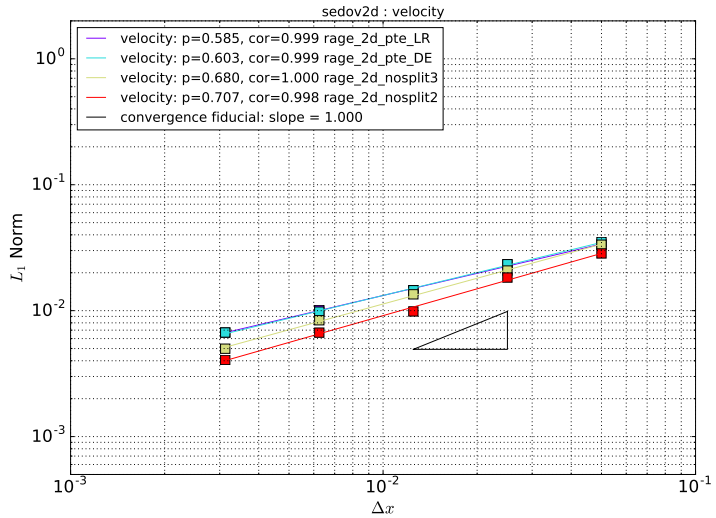


Fig. 53: Convergence analysis for velocity for 2D Sedov for the LR, DE, and UnSplit hydro.

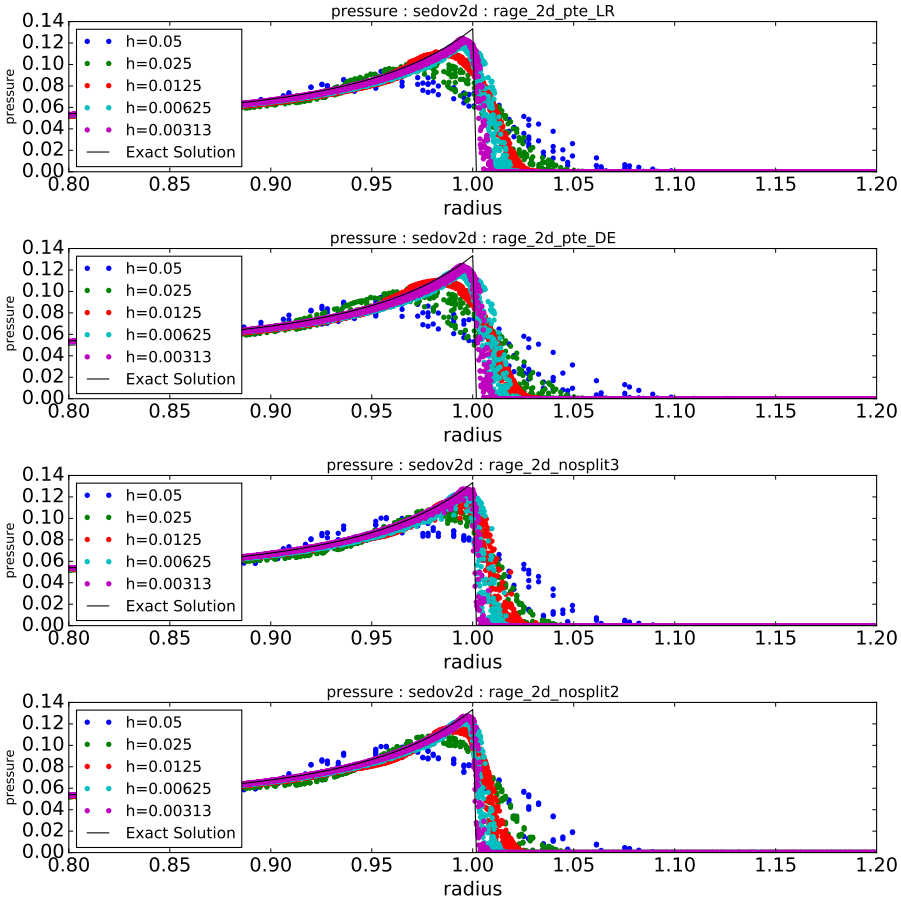


Fig. 54: Pressure for 2D-RZ Sedov with hydro schemes LR, DE, and UnSplit.

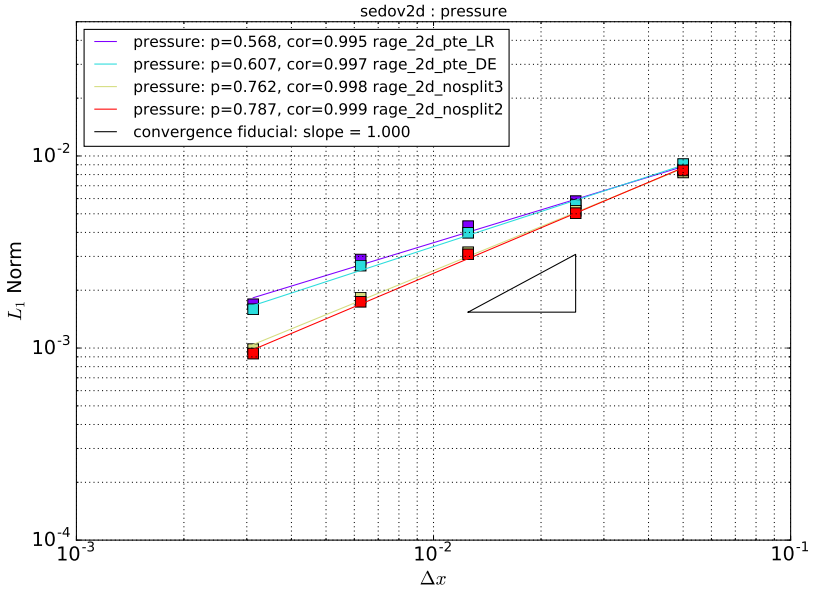


Fig. 55: Convergence analysis for pressure for 2D Sedov for the LR, DE, and UnSplit hydro.

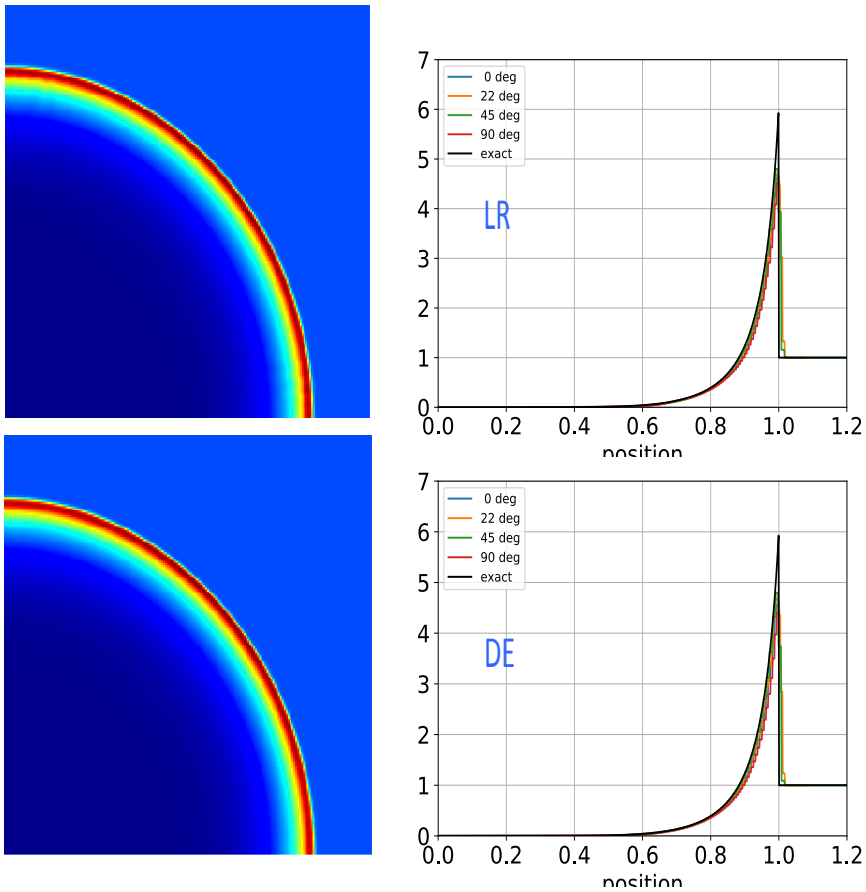


Fig. 56: Density contour plots for 2D Sedov with the LR and DE options, and their corresponding line-out graphs at selected angles in the RZ-plane. The solution resolution is $N = 192$.

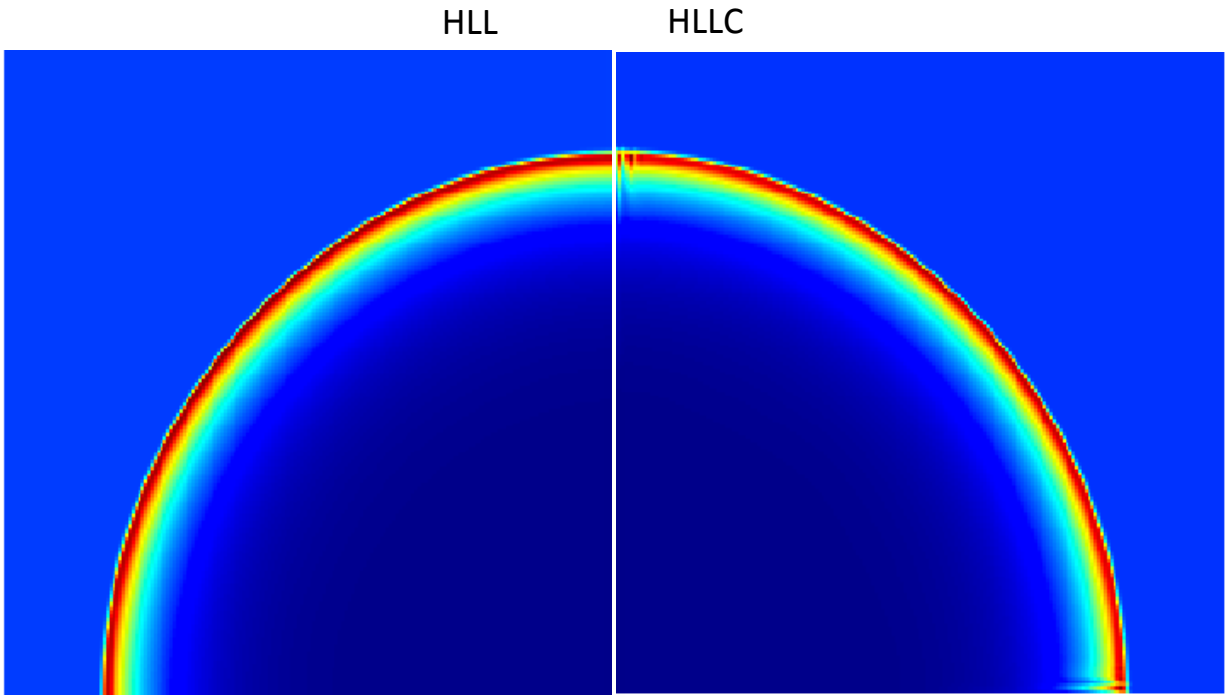


Fig. 57: Density contour plots for 2D Sedov for the UnSplit hydro with HHL and HHLC (left and right panels respectively). The resolution is $N = 192$. The HLLC method (right panel) shows the "carbuncle" instability, a well known and purely numerical instability that occurs when a shock is aligned with the numerical grid.

H. xRAGE Noh Problem

1) Summary of Observations: The results for the 1D Noh runs are illustrated in Figs. 58-63 for the options LR/PTE and DE/PTE. The nonPTE option gives identical results, and the UnSplit hydro does not yet work on the Noh problem. As with Sedov, the convergence analysis for LR and DE uses 8 levels of grid refinement with $N = 24, 48 \cdots 3072$. The first four levels start in the non-asymptotic regime, and then turn over around $N = 192$ to give convergence rates close to $p = 1$.

Figures 64 and 65 illustrate the Noh problem for 2D-RZ geometry with the LR and DE options. For LR, the convergence rate is $p = 0.906$, which we take to be first order. In the lower panel of Fig. 64 for DE, there is a significant amount of scatter in the density plot near the shock, even at small grid refinements, and the DE case does not converge. This is in contrast to the density plot for LR, where the refinements produce better results around the shock. Fig. 65 shows density contour plots for DE and LR, with corresponding line-outs at selected angles. It is clear that DE suffers from significant mesh imprinting, and this explains the scatter seen in Fig. 64. This should be contrasted with DE in 1D, where it is slightly more accurate than LR.

2) Noh in xRAGE, 1D Spherical: Figures 59-63 illustrate the 1D Noh solutions and their convergence analyses for density, velocity, and pressure, respectively, for each option LR and DE. The UnSplit hydro does not yet work on the Noh problem.

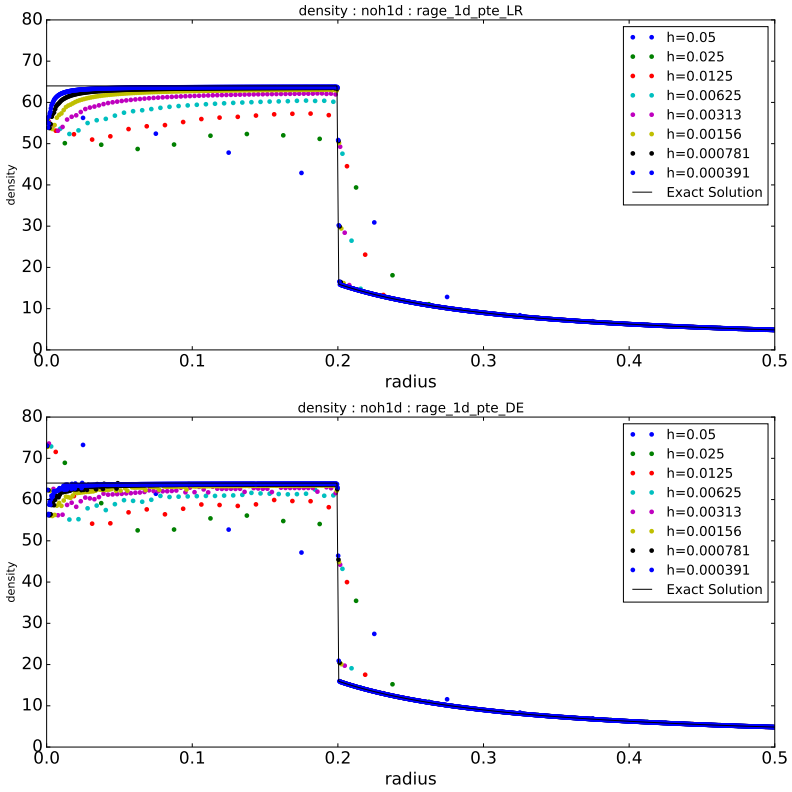


Fig. 58: Solution for density for 1D Noh for the LR and DE.

3) Noh in xRAGE, 2D-RZ: Figures 64 and 65 illustrate the 2D-RZ Noh solutions for LR and DE. The UnSplit hydro does not yet work for the Noh problem.

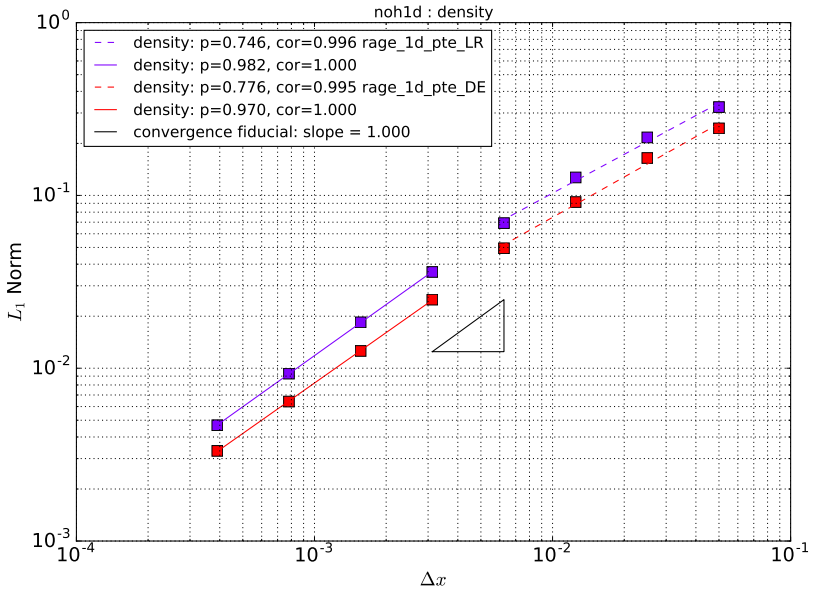


Fig. 59: Convergence analysis for density for 1D Noh for the LR and DE.

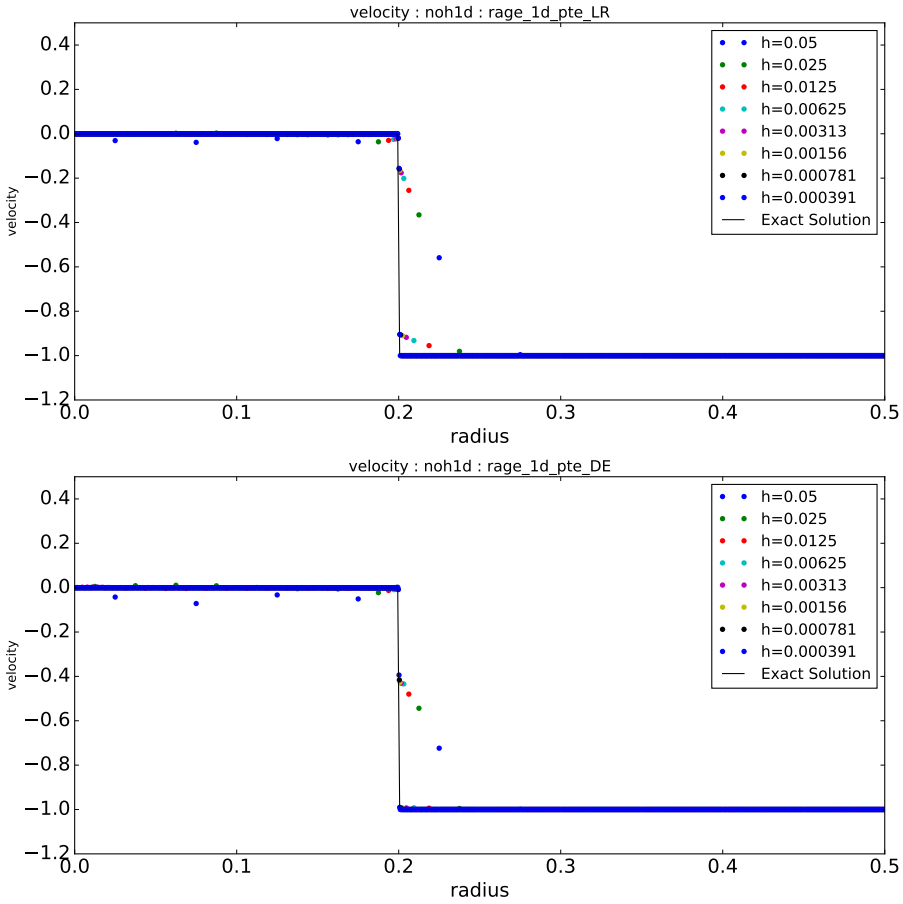


Fig. 60: Solution for velocity for 1D Noh for the LR and DE.

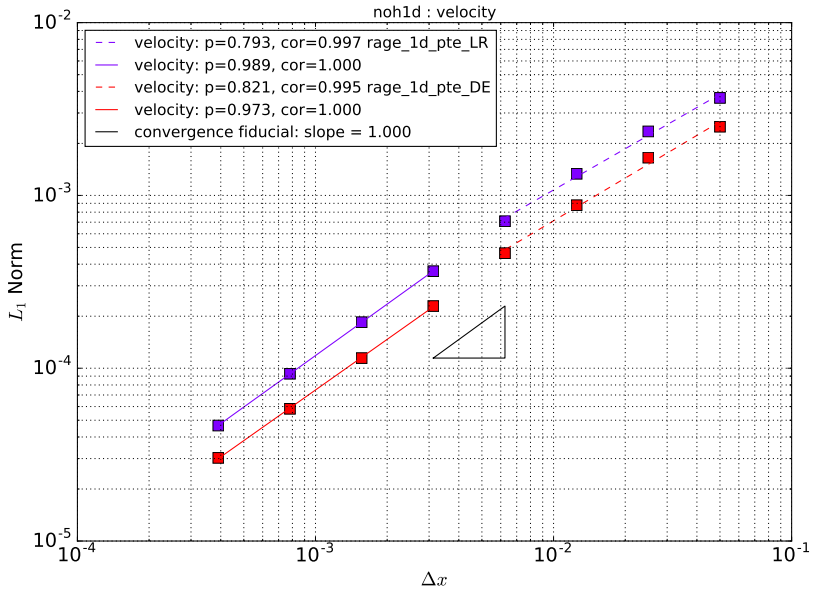


Fig. 61: Convergence analysis for velocity for 1D Noh for the LR and DE.

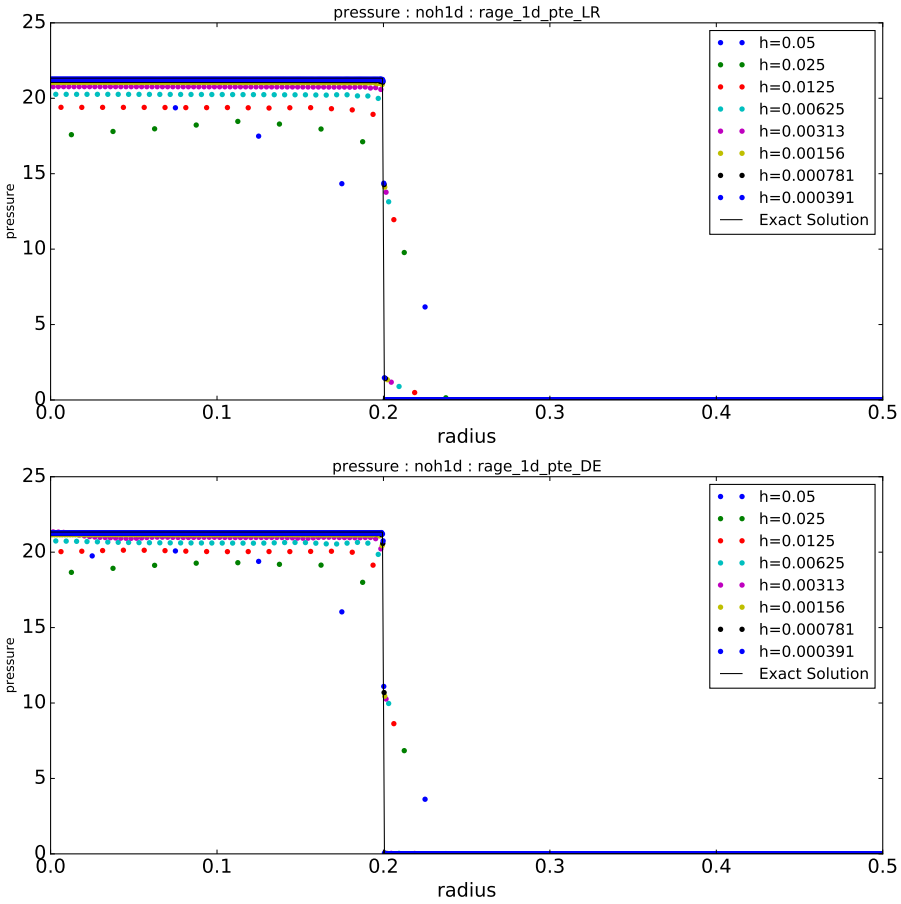


Fig. 62: Solution for pressure for 1D Noh for the LR and DE.

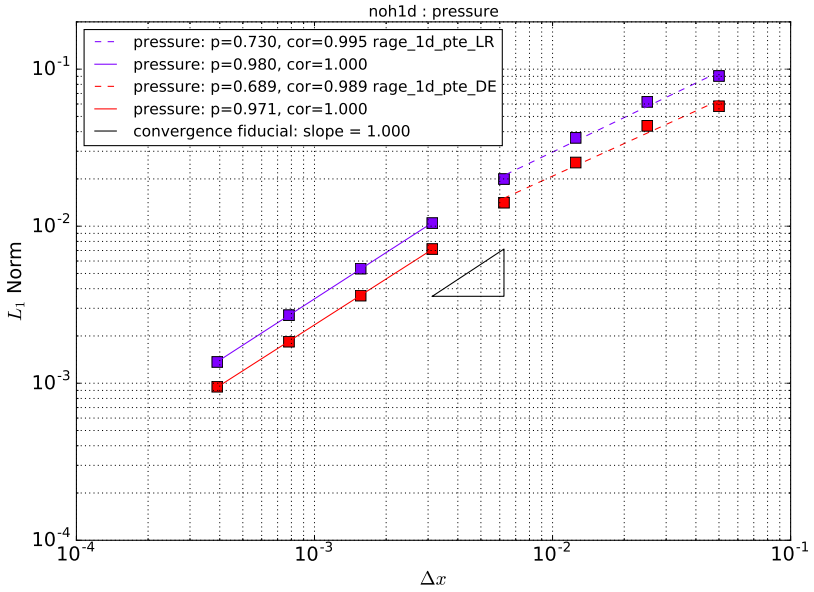


Fig. 63: Convergence analysis for pressure for 1D Noh for the LR and DE.

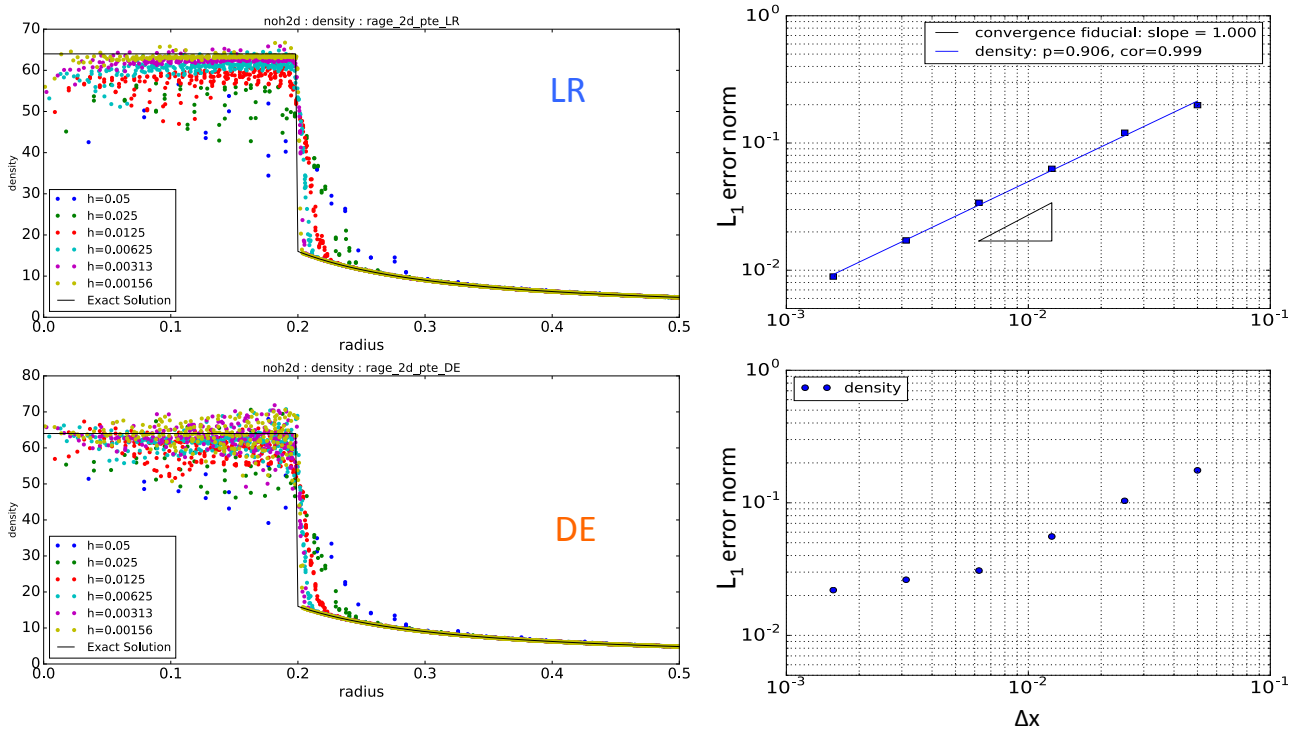


Fig. 64: (left) The density contour plot for 1D-Noh for LR and DE. Note the high degree of scatter for DE as the mesh resolution decreases. (right) Convergence analyses for LR and DE. Note that LR is 1^{st} order convergent, while DE is 0^{th} order convergent.

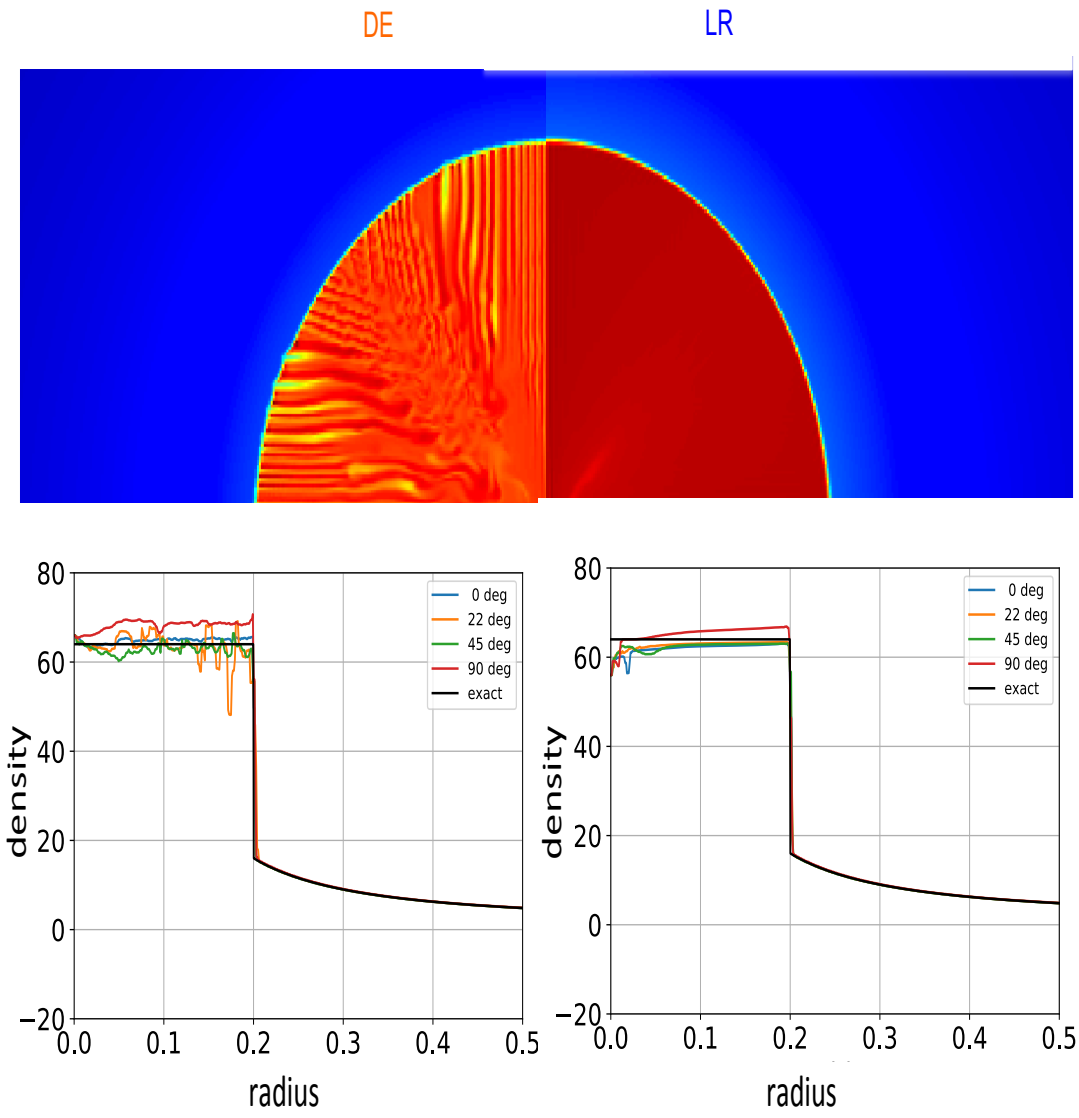


Fig. 65: (top) Density contour plots for 2D-Noh for LR and DE. (bottom) Line-out plots at selected angles.

I. xRAGE Riemann Shock Tube Problems

In this section, we analyze the six Riemann shock tube problems defined in the introduction:

1. Sod
2. Einfeldt
3. Stationary Contact
4. Slow Shock
5. Shock-Contact-Shock
6. LeBlanc

The xRAGE runs have been performed in 1D and 2D Cartesian geometry. Since the 2D results are almost identical to the 1D runs, we only present results in 1D Cartesian geometry. The density profiles and convergence analyses for the various cases are presented in Figs. 66-77. For the Sod problem, we show the density profiles and convergence analyses for all three hydro options LR, DE, and UnSplit.

1) Sod Shock Tube (Riemann Test 1): The Sod problem is the canonical shock tube problem, containing a rarefaction-contact-shock structure. The pressure profiles for LR, DE, and UnSplit are shown in Fig. 66, and the corresponding convergence analyses are presented in Fig. 67. We run this test with 11 levels of refinement, ranging over $N = 50, 100, \cdots, 51200$. Since this problem contains a contact discontinuity, and since xRAGE is a 2^{nd} order accurate code, the asymptotic convergence rate should be $p = 2/3$. However, the observed convergence rates in Fig. 67 are $p \approx 1$ for the first 8 refinements in resolution; the precise values are $p = 0.905, 0.879, 0.811$, for LR, DE, and UnSplit respectively. Only for iterations 9-11 does the convergence rate start to turn over to $p = 0.773, 0.760, 0.811$ (for LR, DE, and UnSplit), which suggests a convergence rate of $p = 2/3$ at fine resolutions.

2) Einfeldt Problem (Riemann Test 2) in xRAGE, 1D Cartesian: Figures 68 and 69 illustrate the Einfeldt problem, which consists of two separating strong rarefaction waves, with a near-vacuum between them. We include only the results of the LR hydro scheme in this and the following Riemann problems, as DE is qualitatively similar. Also, the xRAGE simulations are performed over 8 refinements in resolution, $N = 50, 100 \cdots 6400$. Since this problem does not contain a contact discontinuity, and since the profile is C^1 continuous, we therefore expect a convergence rate of $p = 1$. As seen in Fig. 69, however, the observed rate is only $p = 0.783$, and we see that xRAGE is performing worse than expected for this problem at these resolutions.

3) Stationary Contact Problem (Riemann Test 3) in xRAGE, 1D Cartesian: Figures 70 and 71 illustrate the stationary contact problem. This problem consists of a strong shock wave moving to the right, a stationary contact, and a strong rarefaction wave moving to the left. At 8 levels of refinement, we observe that the convergence rate is $p \approx 1$. There appears to be an oscillatory behavior in the L_1-norm about the $p = 1$ line, and we expect that the convergence should approach $p = 2/3$ with more grid refinement.

4) Slow Shock Problem (Riemann Test 4) in xRAGE, 1D Cartesian: Figures 72 and 73 illustrate the slow shock problem. This problem consists of a Mach 3 shock wave moving slowly to the right. For the first four resolution refinements, the convergence rate in Fig. 73 is observed to $p = 1.153$. At the 5^{th} resolution, value of p levels off and then drops at a rate $p < 1$. This is at the same resolution at which the FLAG results diverge from their expected scaling behavior.

5) Shock-Contact-Shock Problem (Riemann Test 5) in xRAGE, 1D Cartesian: Figures 74 and 75 illustrate the shock-contact-shock problem. In this problem, two shocks separate from the initial state, with a contact between them. This produces errors in all the fields, and this problem tests an algorithm's ability to deal with these errors. Linear with $p = 0.926$ over 8 levels of refinement. We interpret this to be 1^{st} order convergent over the grid resolutions show, although because of the contact discontinuity, we expect $p = 2/3$ at higher resolution.

6) LeBlanc Problem (Riemann Test 6) in xRAGE, 1D Cartesian: Figures 76 and 77 illustrate the LeBlanc problem. This problem is a strong shock and strong rarefaction version of the basic rarefaction-contact-shock problem, or a strongly shocked version of the Sod problem. The convergence rate starts out around $p = 0.75$, and changes to $p \approx 1$ on the 5^{th} resolution, and we expect $p = 2/3$ behavior at even higher resolution.

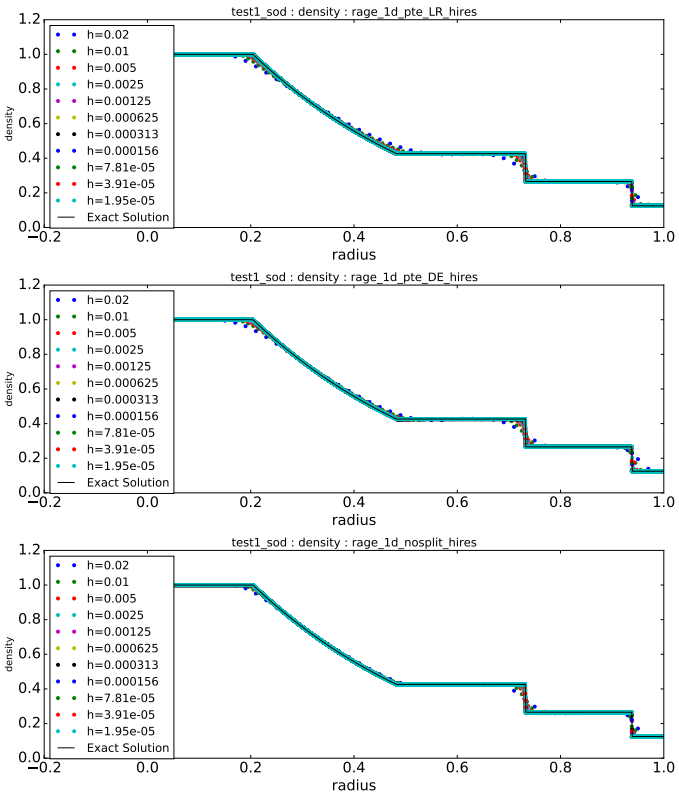


Fig. 66: The density profile for the Sod problem (Riemann Test 1) for LR, DE, and the UnSplit hydro methods.

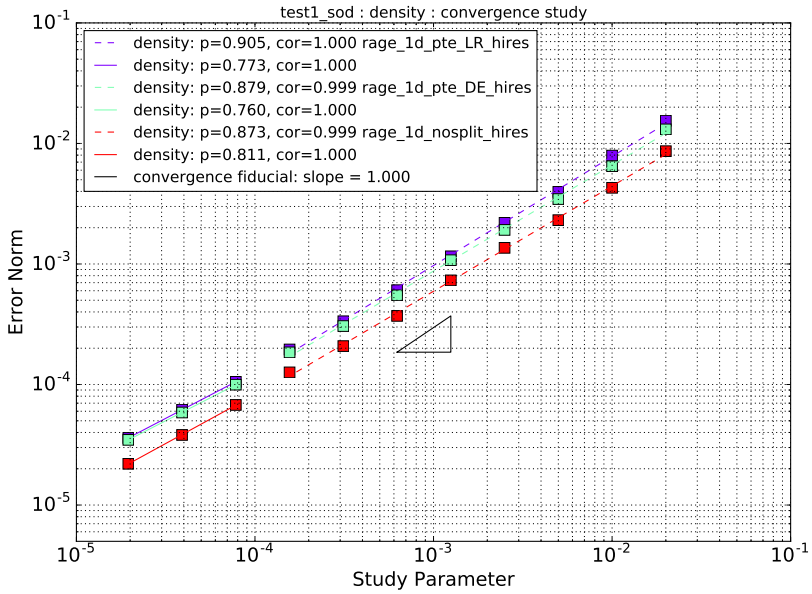


Fig. 67: The convergence analyses for the Sod problem above.

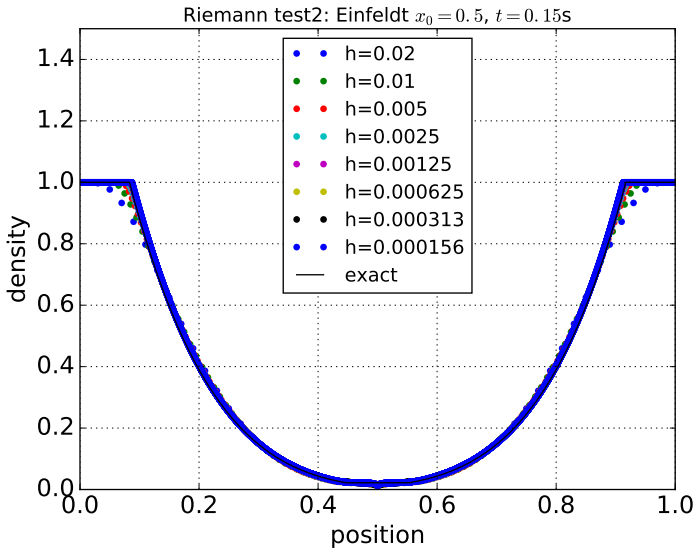


Fig. 68: The density profile for the Einfeldt problem (Riemann Test 2) for the hydro option LR.

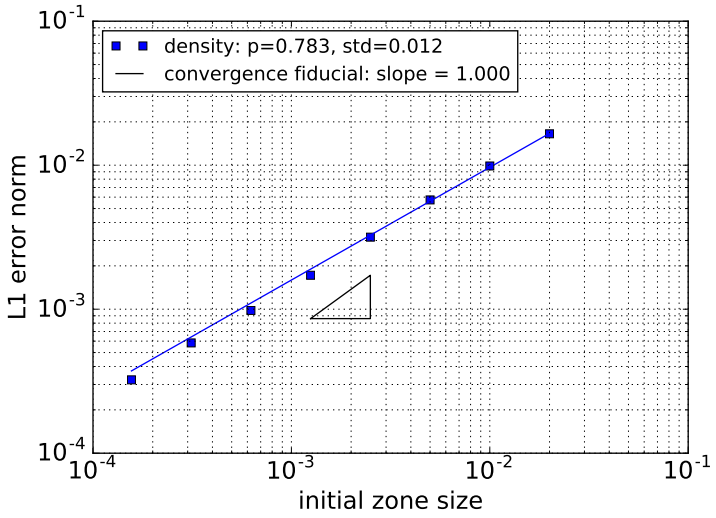


Fig. 69: The convergence analysis for the Einfeldt problem above.

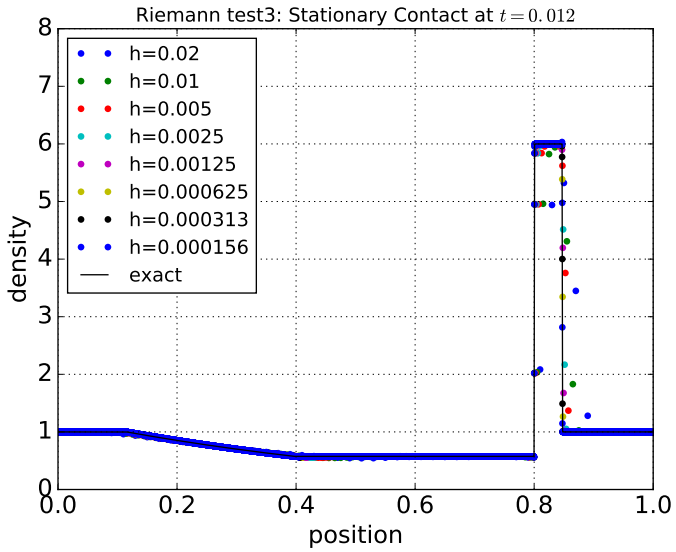


Fig. 70: The density profile for the stationary contact problem (Riemann Test 3) for the hydro option LR.

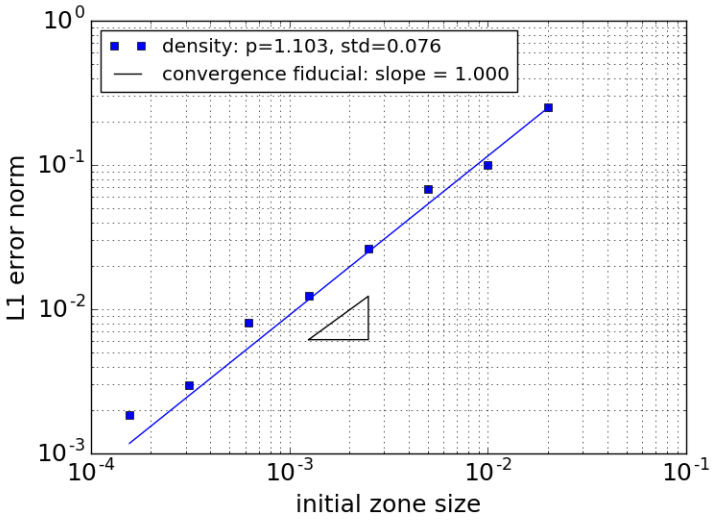


Fig. 71: The convergence analysis for the stationary contact problem above.

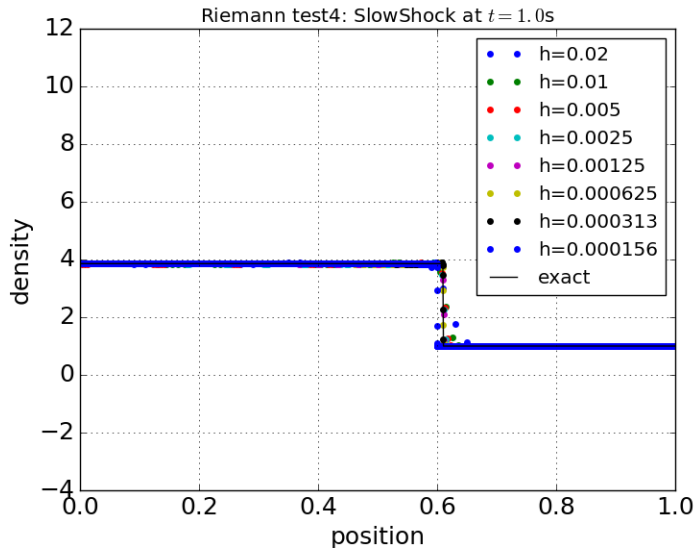


Fig. 72: The density profile for the slow shock problem (Riemann Test 4) for the hydro option LR.

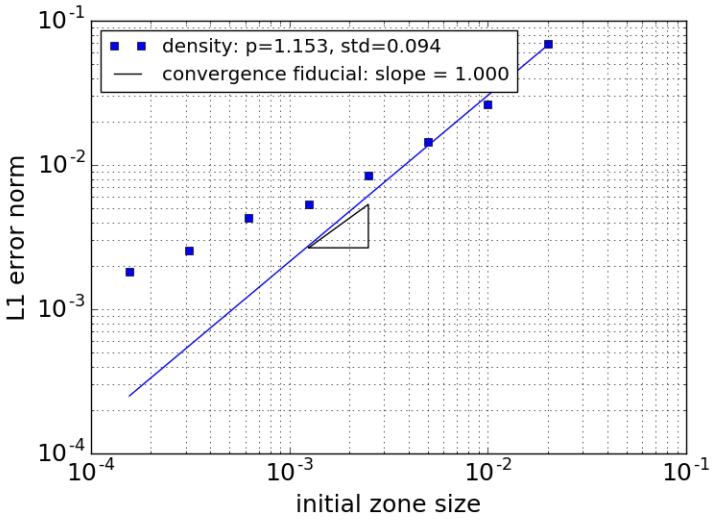


Fig. 73: The convergence analysis for the slow shock problem above.

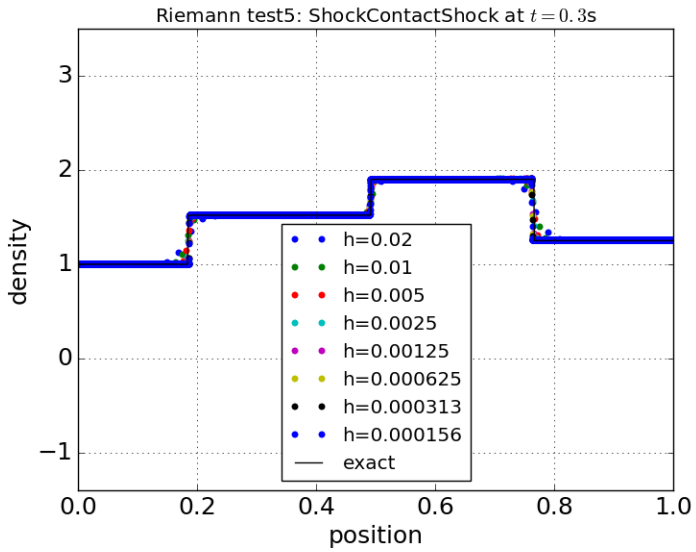


Fig. 74: The density profile for the slow shock-contact-shock problem (Riemann Test 5) for the hydro option LR.

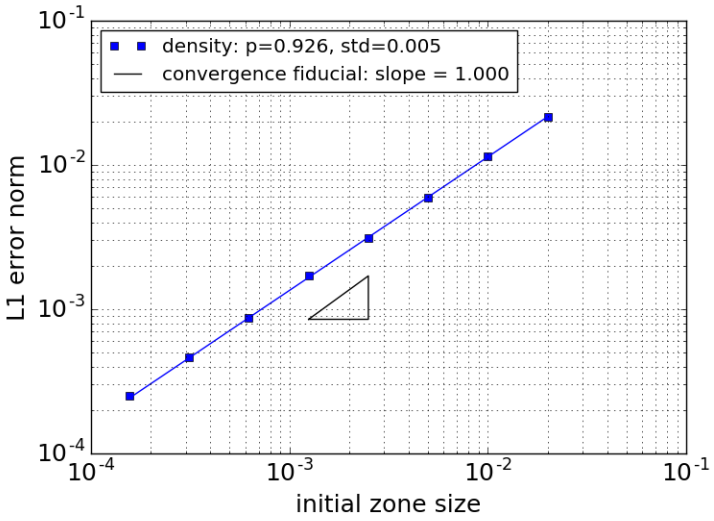


Fig. 75: The convergence analysis for the shock-contact-shock problem above.

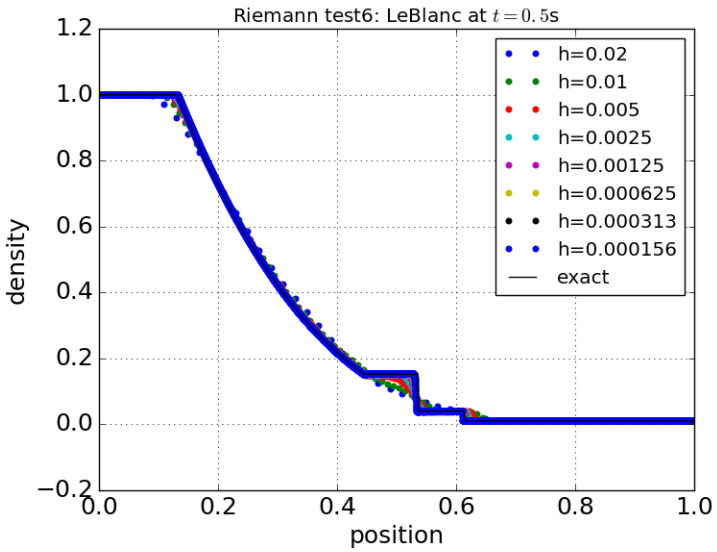


Fig. 76: The density profile for the LeBlanc problem (Riemann Test 6) for the hydro option LR.

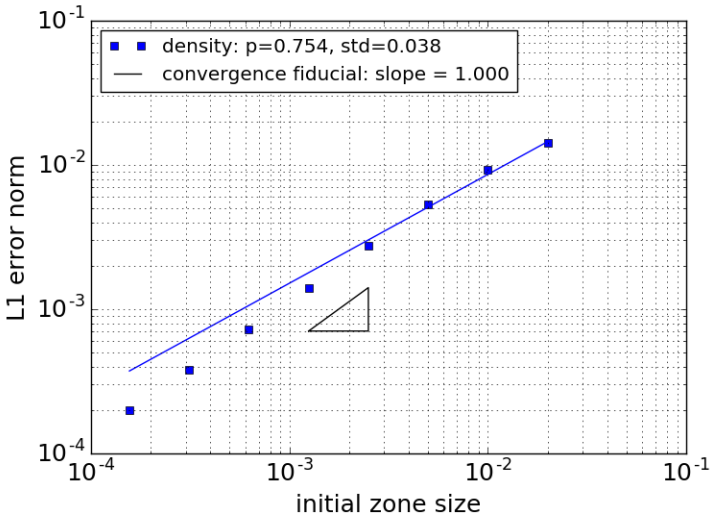


Fig. 77: The convergence analysis for the LeBlanc problem above.

J. xRAGE Noh2 Problem

1) Summary of Observations: This is the uniform collapse problem, or Noh's second problem (which we refer to as Noh2). In this variant of the Noh problem, the initial velocity profile is taken to be $u = -r$ (rather than $u = -1$), The Noh2 solution is continuos and contains no shocks.; therefore, the numerical solution is expected to be 2^{nd} order convergent. The velocity profiles and convergence analyses for LR and DE are shown in Fig. 78. Surprisingly, Noh2 converges at 1^{st} order. This could indicate a problem with the limiter scheme in xRAGE, or a problem with the way in which the $u = -r$ condition has been implemented. Further work is needed to resolve this issue.

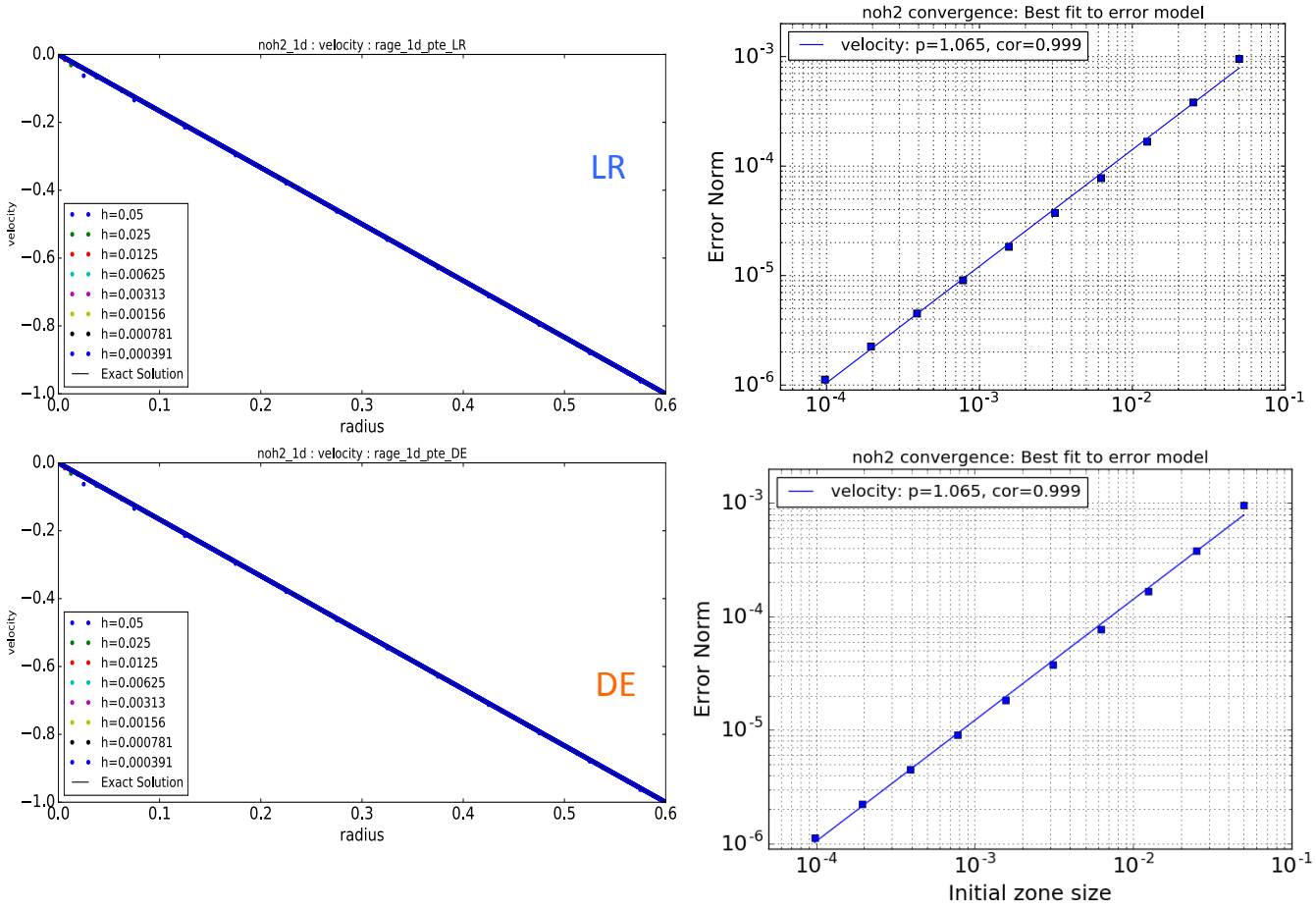


Fig. 78: (left) Velocity profile for 1D Noh2 LR and DE. (right) Corresponding convergence analyses.

V. Verification of Radiative Shocks and an Ion-electron Shock

This section documents five test problems that were developed for ExactPack, two of which were used as code-verification tests for xRAGE as a part of this L2 Milestone effort. All of these solutions are relatively new, and only the nonequilibrium-diffusion solution has been used as a convergence analysis tool in a publication [7]. Some of the other solutions have been used to verify radiation hydrodynamic codes, but typically as a comparison in the "eyeball" norm. All five test problems are time-independent, 1D planar, multi-physics shock problems, one being a plasma ("ion-electron") shock [23] and the other four being radiative shocks. The radiative-shock problems exercise frequency-independent ("grey") radiation models for equilibrium-diffusion [24], nonequilibrium-diffusion [25], and S_n-transport radiation [5]. These separate models represent three of the four radiative-shock problems. The fourth radiative-shock problem [6] tests whether an algorithm is "asymptotic preserving", for which an algorithm is capable of producing either a high-order solution (i.e., nonequilibrium-diffusion, here) or a low-order solution (i.e., equilibrium-diffusion) at a low computational cost.

As a matter of interest regarding the mathematical development and the code development of these five solutions, four involve solving a two-point boundary-value problem, where the two boundaries are in equilibrium at "infinity". These are the nonequilibrium-diffusion shock problem, the asymptotic-preserving radiative-shock problem, the S_n-transport shock problem, and the ion-electron shock problem. As such, their mathematical solution method and their code architecture are quite similar, and only the equations being solved must be changed. In contradistinction, the equilibrium-diffusion solution is a one-point boundary-value problem, and so requires its own, albeit short, solution method. The solution method implemented for this set of two-point boundary-value problems is described in the paper documenting the grey S_n-transport radiative-shock solutions [5].

Since all of these solutions are time independent and 1D planar they are also shift invariant. We take advantage of this shift invariance when running these problems in xRAGE by meshing a long shocktube, initializing xRAGE with the semi-analytic solution, and allowing xRAGE to propagate the solution for 10^{-7} seconds. To prevent the spurious shedding of waves from affecting xRAGE's results, we use freeze boundary conditions in the two regions at the end of the shocktube. We expect xRAGE to accomplish at least two fundamental tasks while propagating the radiative-shock solution: 1) first, that xRAGE will not change the shape of the solution, and 2) second, that xRAGE will propagate the solution in a manner that would be expected from Galilean invariance.

The rest of this section is organized as follows: in subsection V-A we provide a brief introduction to the full set of equations describing radiation hydrodynamics (RH); subsection V-B presents the equilibrium-diffusion approximation, the modified RH equations, a radiative-shock solution to those equations and xRAGE convergence analysis results using that solution; subsection V-C presents the nonequilibrium-diffusion approximation, the modified RH equations, two radiative-shock solutions to those equations and the xRAGE convergence analyses using those solutions; subsection V-D presents the asymptotic variables, the modified (nonequilibrium-diffusion) RH equations and results showing that nonequilibrium-diffusion radiative-shocks solutions are asymptotic to equilibrium-diffusion radiative-shock solutions; subsection V-E presents the S_n-transport approximation, the modified RH equations, and two radiative-shock solutions.

A. Introduction to radiation hydrodynamics

Radiation hydrodynamics describes physical environments in which the radiation affects the physical characteristics of the material flow field. As an example, the radiation may alter the temperature field of the fluid, or, if the radiation momentum is at least comparable to the fluid momentum, then the velocity field of the fluid may be significantly altered. Of course, if the fluid is sufficiently transparent to the radiation, then it is possible that the radiation and fluid may inhabit the same physical space without interacting. This coupling between the fluid and the radiation is determined by the interaction mean-free-path, λ, or similarly, the interaction cross section, $\sigma = 1/\lambda$. In most physical and astrophysical applications of interest, the radiation interaction mean-free-path is much larger than the mean-free-path between any set of fluid particles, so that Eulerian hydrodynamics is sufficient to describe the fluid. As the name "radiation hydrodynamics" implies, we seek to couple radiation to a hydrodynamical model, and to model the interactions between the radiation and the fluid. Since radiation is massless there is nothing to couple to the mass conservation equation. However, radiation momentum and radiation energy sources do need to be coupled to the Euler equations:

$$\partial_t \rho + \partial_i \left(\rho\, u_i\right) = 0\,, \tag{36a}$$

$$\partial_t \left(\rho\, u_i\right) + \partial_j \left(\rho\, u_i\, u_j + p_{ij}\right) = -S_{i,rp}\,, \tag{36b}$$

$$\partial_t \left(\frac{1}{2}\rho\, u^2 + \rho\, e\right) + \partial_j \left[u_i \left(\frac{1}{2}\rho\, u_i\, u_j + \rho\, e\, \delta_{ij} + p_{ij}\right)\right] = -S_{re}\,, \tag{36c}$$

$$\frac{1}{c}\partial_t I_\nu\left(\Omega\right) + \Omega_i\, \partial_i I_\nu\left(\Omega\right) = Q_\nu\left(\Omega\right)\,, \tag{36d}$$

$$Q_\nu\left(\Omega\right) = -\sigma_{t,\nu}\, I_\nu\left(\Omega\right) + \sigma_{a,\nu} B_\nu + \frac{\sigma_{s,\nu}}{4\pi}\int_{4\pi} I_\nu\left(\Omega'\right) d\Omega'\,, \tag{36e}$$

$$S_{re} = \int_0^\infty \int_{4\pi} Q_\nu\left(\Omega\right)\, d\Omega\, d\nu = \partial_t \mathcal{E} + \partial_i \mathcal{F}_i\,, \tag{36f}$$

$$S_{rp,i} = \frac{1}{c}\int_0^\infty \int_{4\pi} \Omega_i\, Q_\nu\left(\Omega\right)\, d\Omega\, d\nu = \frac{1}{c^2}\partial_t \mathcal{F}_i + \partial_j \mathcal{P}_{ij}\,, \tag{36g}$$

$$\mathcal{E} = \frac{1}{c}\int_0^\infty \int_{4\pi} I_\nu\left(\Omega\right)\, d\Omega\, d\nu\,, \tag{36h}$$

$$\mathcal{F}_i = \int_0^\infty \int_{4\pi} \Omega_i\, I_\nu\left(\Omega\right)\, d\Omega\, d\nu\,, \tag{36i}$$

$$\mathcal{P}_{ij} = \frac{1}{c}\int_0^\infty \int_{4\pi} \Omega_i\, \Omega_j\, I_\nu\left(\Omega\right)\, d\Omega\, d\nu\,. \tag{36j}$$

The first three equations are the conservation statements of mass, total momentum and total energy, the fourth and fifth equations are the angle- and frequency-dependent radiation transport equation, and the last five equations are angular moments of either the radiation source, $Q_\nu(\Omega)$, which gives the radiation energy and momentum sources, S_{re} and $S_{rp,i}$, or the radiation intensity, $I_\nu(\Omega)$, which gives the radiation energy density, $\mathcal{E}$, the radiation flux, $\mathcal{F}_i$, and the radiation pressure, $\mathcal{P}_{ij}$. A detailed exposition may be found in the book by Mihalas and Mihalas [26]. These equations represent the nonrelativistic model of radiation hydrodynamics, and otherwise contain few other model approximations beyond geometric optics, binary interactions, and the lack of quantum corrections. Despite these basic and standard assumptions these equations are still expensive to solve computationally and many models have been developed to aide their solution (e.g., non/equilibrium-diffusion and S_n-transport). However, since all of the solutions contained in this section are grey, 1D planar, and time-independent, we modify these equations accordingly:

$$\partial_x\left(\rho\, u\right) = 0\,, \tag{37a}$$

$$\partial_x\left(\rho\, u^2 + p\right) = -S_{rp}\,, \tag{37b}$$

$$\partial_x\left[u\left(\frac{1}{2}\rho\, u^2 + \rho\, e + p\right)\right] = -S_{re}\,, \tag{37c}$$

$$\mu\, \partial_x I\left(\mu\right) = Q\left(\mu\right)\,, \tag{37d}$$

$$Q\left(\mu\right) = -\sigma_t\, I\left(\mu\right) + \sigma_a \frac{a_r\, c\, T^4}{4\pi} + \frac{\sigma_s}{4\pi}\int_{4\pi} I\left(\mu'\right) d\mu'\,, \tag{37e}$$

$$S_{re} = 2\,\pi \int_{-1}^{1} Q\left(\mu\right)\, d\mu = \partial_x \mathcal{F}\,, \tag{37f}$$

$$S_{rp} = \frac{2\,\pi}{c}\int_{-1}^{1} \mu\, Q\left(\mu\right)\, d\mu = \partial_x \mathcal{P}\,, \tag{37g}$$

$$\mathcal{E} = \frac{2\,\pi}{c}\int_{-1}^{1} I\left(\mu\right)\, d\mu\,, \tag{37h}$$

$$\mathcal{F}_i = 2\,\pi \int_{-1}^{1} \mu\, I\left(\mu\right)\, d\mu\,, \tag{37i}$$

$$\mathcal{P}_{ij} = \frac{2\,\pi}{c}\int_{-1}^{1} \mu^2\, I\left(\mu\right)\, d\mu\,. \tag{37j}$$

In the next subsection we introduce the equilibrium-diffusion approximation. It represents the strongest set of assumptions that are to be imposed on the RH equations to be considered in this section. Subsequent subsections will relax these assumptions, until the S_n-transport approximation is presented in subsection V-E whose equations look similar to equations (37).

B. The equilibrium-diffusion radiation model

The equilibrium-diffusion radiation model was introduced by Eddington [27] for the purposes of rigorously studying the interior of stars. In addition to Eulerian hydrodynamics, it assumes [26]

1) that the fluid-radiation system is in thermal equilibrium,
2) that the radiation pressure is isotropic, and
3) that the mean-free-path for interactions between the fluid and radiation in the system is much smaller than the system's size so that it may be reasonably assumed that the radiation diffuses through the system.

The effect of the thermal-equilibrium assumption is that the radiation energy density is proportional to the fourth-power of the material temperature, $\mathcal{E} = a_r\, T^4$. Isotropy of the radiation pressure is equivalent to the statement that the radiation pressure is one-third the radiation energy density, $\mathcal{P} = \mathcal{E}/3$, which is often referred to as the Eddington approximation. The diffusion

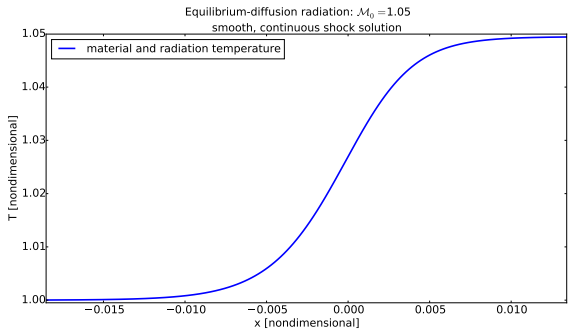
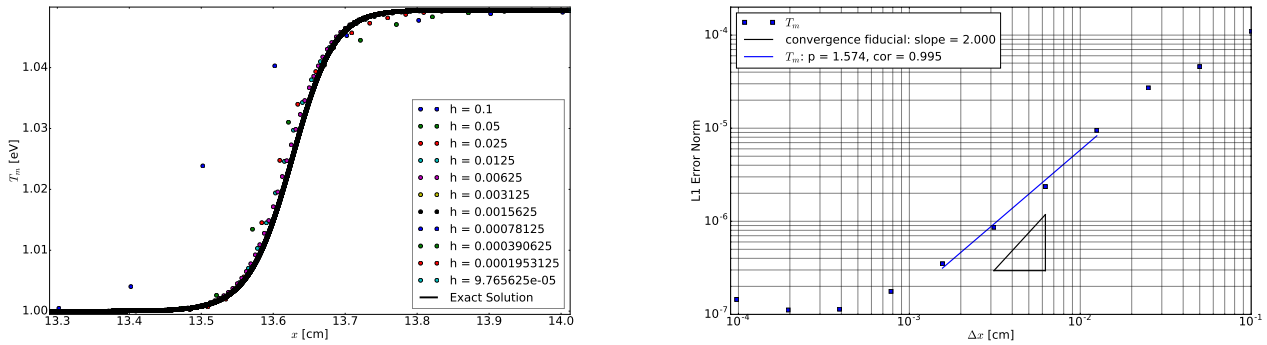


Fig. 79: The equilibrium-diffusion radiative-shock solution for $\mathcal{M}_0 = 1.05$.



(a) Refinement of the xRAGE mesh considerably improves agreement with the solution.

(b) The L1 error norm of the xRAGE solutions compared to the semi-analytic solution.

Fig. 80: The equilibrium-diffusion radiative-shock solution for $\mathcal{M}_0 = 1.05$, and results from xRAGE propagating the solution down the shocktube.

approximation implies that the radiation flux is proportional to the negative gradient of the radiation energy density. In the lab-frame, where the conservation equations can be defined, the radiation enthalpy is added to the negative gradient so the radiation flux is, $\mathcal{F} = -\left[c/(3\,\sigma_t)\right]\partial_x \mathcal{E} + 4\,u\,\mathcal{E}/3$. Interestingly, in the equilibrium-diffusion approximation the radiation intensity depends directly on these radiation variables, $I = (c\,\mathcal{E} + \mu\,\mathcal{F})/4\pi$.

In the equilibrium-diffusion approximation the RH equations are:

$$\partial_x\left(\rho\,u\right) = 0\,, \tag{38a}$$

$$\partial_x\left(\rho\,u^2 + p + \frac{a_r\,T^4}{3}\right) = 0\,, \tag{38b}$$

$$\partial_x\left[u\left(\frac{1}{2}\rho\,u^2 + \rho\,e + p + \frac{4}{3}\,a_r\,T^4\right)\right] = \frac{4\,a_r\,c}{3}\partial_x\left(\frac{T^3}{\sigma_t}\partial_x T\right)\,. \tag{38c}$$

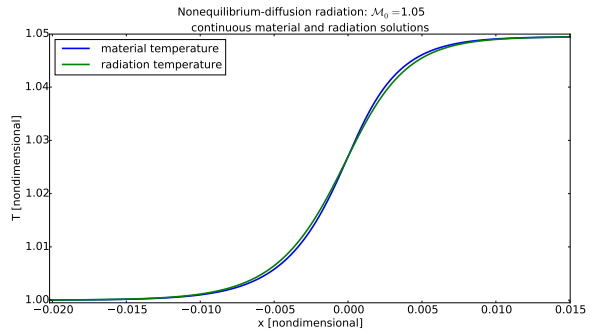
A quick inspection of these equations shows that they depend strictly on material variables. However, as such, much of the radiation specific content has been lost. Further, just as the Euler equations for hydrodynamics contain shock solutions, these equations admit radiative-shock solutions. One of the important physical results obtained from radiative-shock solutions in the equilibrium-diffusion approximation is that the radiation smooths out the shock structure and it is possible to have a shock solution with a smooth, continuous profile, as shown in Figure 79.

The results of running this problem in xRAGE are shown in Figures 80. The figure on the left provides an eyeball comparison of the improvement obtained by going to higher resolutions. The figure on the right shows that xRAGE obtains approximately 3/2-order convergence. At higher resolutions the xRAGE L1 error norm values flatten out around 10^{-7}. It is expected that this is due to a tolerance setting of an iterative solver.

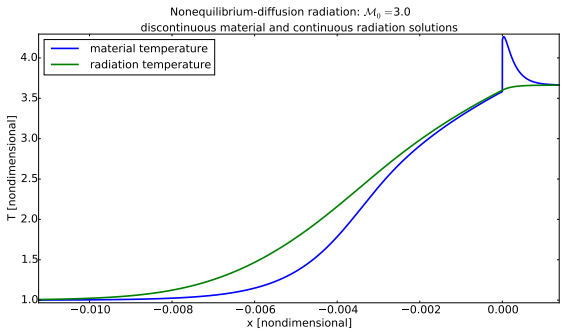
C. The nonequilibrium-diffusion radiation model

As a reminder, the equilibrium-diffusion approximation assumes, in addition to Eulerian hydrodynamics, [26]

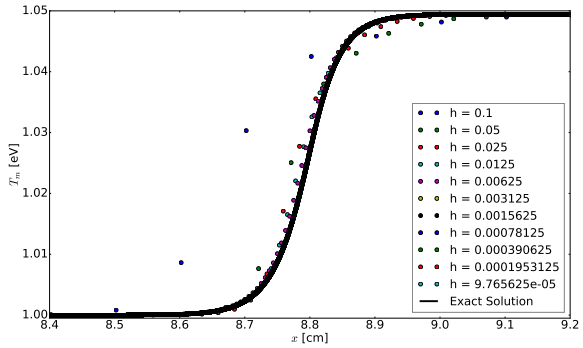
1) that the fluid-radiation system is in thermal equilibrium,
2) that the radiation pressure is isotropic, and



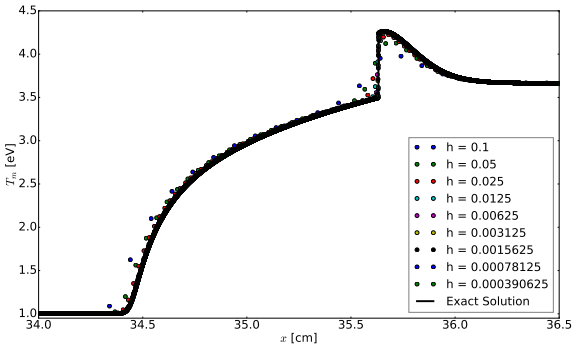
(a) The nonequilibrium-diffusion radiative-shock solution for $\mathcal{M}_0 = 1.05$.



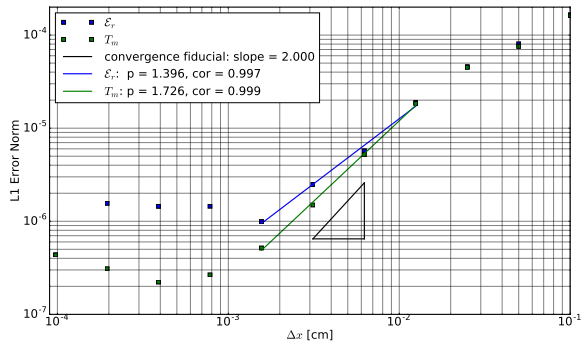
(b) The nonequilibrium-diffusion radiative-shock solution for $\mathcal{M}_0 = 3$.



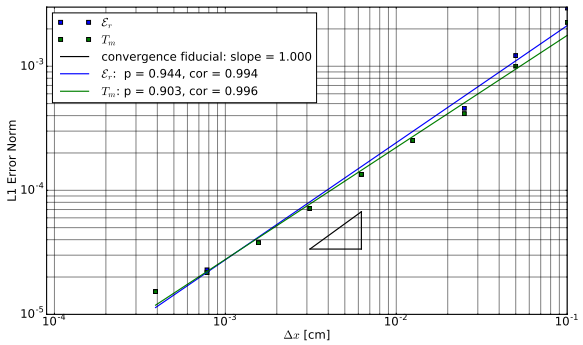
(c) Refinement of the xRAGE mesh considerably improves agreement with the solution.



(d) Refinement of the xRAGE mesh considerably improves agreement with the solution.



(e) The L1 error norm of the xRAGE solutions compared to the semi-analytic solution.



(f) The L1 error norm of the xRAGE solutions compared to the semi-analytic solution.

Fig. 81: Nonequilibrium-diffusion radiative-shock solutions for $\mathcal{M}_0 = 1.05$, in the left column, and $\mathcal{M}_0 = 3$, in the right column, along with results from xRAGE propagating the solution down the shocktube.

3) that the mean-free-path for interactions between the fluid and radiation in the system is much smaller than the system's size so that it may be reasonably assumed that the radiation diffuses through the system.

The nonequilibrium-diffusion approximation relaxes the first assumption and allows the fluid to have a material temperature, T, and the radiation to have a separate temperature, θ. Now, the radiation energy density is defined with respect to the radiation temperature, $\mathcal{E} = a_r\,\theta^4$. Isotropy of the radiation pressure is still equivalent to the statement that the radiation pressure is one-third the radiation energy density, $\mathcal{P} = \mathcal{E}/3$. The diffusion approximation still implies that the radiation flux is proportional to the negative gradient of the radiation energy density, so in the lab-frame, the radiation flux is still, $\mathcal{F} = -[c/(3\,\sigma_t)]\,\partial_x\mathcal{E} + u\,(\sigma_t\,\mathcal{E}/3 + \sigma_s\,\mathcal{E} + \sigma_a\,a_r\,T^4)$. Finally, the radiation intensity also still depends directly on these radiation variables, $I = (c\,\mathcal{E} + \mu\,\mathcal{F})/4\pi$.

In the nonequilibrium-diffusion approximation the RH equations are:

$$\partial_x\,(\rho\,u) = 0\,, \tag{39a}$$

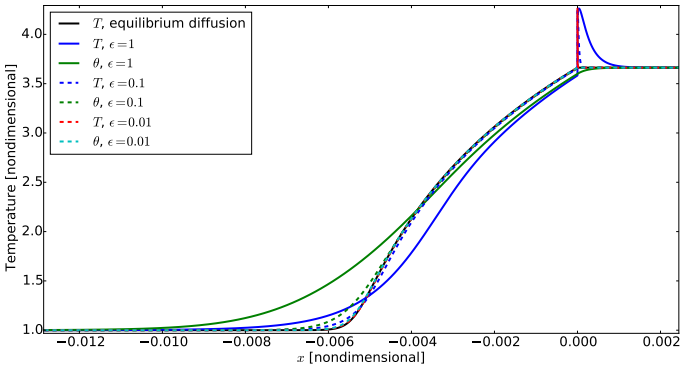


Fig. 82: The asymptotic-preserving (AP) radiative-shock solutions. The equilibrium-diffusion solution is provided as a reference and is the solution which should be produced in the limit that $\epsilon \to 1$. The AP solutions are produced using the nonequilibrium-diffusion radiative-shock solver for the asymptotically small values of $\epsilon = 0.1$ and $\epsilon = 0.01$. The lines for the $\epsilon = 0.01$ AP solutions (red and cyan) lie almost entirely on the line for the equilibrium-diffusion solution (black). It is interesting to see that the Zel'dovich temperature spike is retained in the asymptotic limit.

$$\partial_x \left(\rho\, u^2 + p + \frac{a_r\, \theta^4}{3} \right) = 0\,, \tag{39b}$$

$$\partial_x \left[u \left(\frac{1}{2} \rho\, u^2 + \rho\, e + p + \frac{4}{3}\, a_r\, \theta^4 \right) \right] = \frac{4\, a_r\, c}{3} \partial_x \left(\frac{\theta^3}{\sigma_t} \partial_x \theta \right)\,. \tag{39c}$$

Since we have introduced the radiation temperature into these equations, which is a new variable, we need one more equation in order for the system to be solvable. For this purpose, we introduce the internal energy equation, with material internal energy coupled to the radiation internal energy, $S_{rie} = S_{re} - u\, S_{rp}$:

$$\rho\, u\, \partial_x e + p\, \partial_x u = -S_{rie} = \sigma_a\, c\, \left(\mathcal{E} - a_r\, T^4 \right) - 2\, \sigma_a\, \frac{u}{c}\, \mathcal{F} + \frac{u^2}{c}\, \left(\sigma_t\, \mathcal{P} + \sigma_s\, \mathcal{E} + \sigma_a\, a_r\, T^4 \right)\,. \tag{39d}$$

In the equilibrium-diffusion approximation, discussed in the previous subsection, the right-hand side of this equation evaluates to zero. There, the first term is identically zero due to the thermal equilibrium assumption, whereas the second and third terms are of second order in small parameters and therefore neglected. Restoring this equation restores some of the radiation content that was previously lost.

Smooth radiative-shock solutions are retained in the nonequilibrium-diffusion approximation, as shown in Figure 81a. Figures 81c and 81d present xRAGE convergence results for this smooth, continuous shock. In the asymptotic regime, Figure 81d shows that the radiation energy density is converging at an order of 1.4, while the material temperature is converging at an order of 1.7. At higher resolutions where the error norm is near 10^{-6} the xRAGE results appear to flatten out. It is expected that this is due to a tolerance setting for an iterative solver.

An important physical result obtained from nonequilibrium-diffusion radiative-shock solutions is that the fluid may retain an embedded hydrodynamic shock structure located between the shock's equilibrium boundaries, as shown in Figure 81b. Figures 81e and 81f present xRAGE convergence results for this problem. Figure 81f shows that the radiation energy density and the material temperature both converge at first order. It is impressive that this first-order convergence occurs over two orders of magnitude in resolution space.

D. The asymptotic-preserving radiative-shock problem

Asymptotic-preserving (AP) test problems are rare in the literature. This is because AP problems require two separate but similar problems, one of which is considered a low-order model and the other is, comparatively, a high-order model. Certain specific variables in the high-order model may be scaled (multiplied or divided) by an asymptotically small parameter, $\epsilon \ll 1$, such that when $\epsilon \to 0$ the low-order model is obtained through some order in ϵ. As regards RH, equilibrium-diffusion is the low-order model and nonequilibrium-diffusion is the high-order model. The following variables and their scalings define the equilibrium-diffusion limit [6], for which the the equilibrium-diffusion approximation is exact through first-order in ϵ:

$$\partial_t \to \epsilon^2\, \partial_t\,, \quad \partial_x \to \epsilon\, \partial_x\,, \quad \frac{u}{c} \to \epsilon\, \frac{u}{c}\,, \quad \sigma_s \to \epsilon\, \sigma_s\,. \tag{40}$$

Applying these scalings to the nonequilibrium-diffusion RH equations (39) produces:

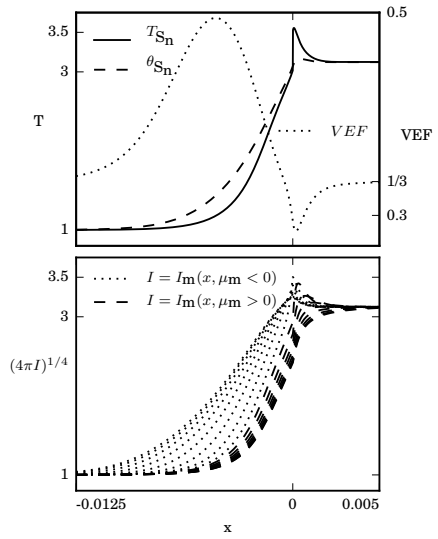


Fig. 83: The S_n-transport radiative-shock solution for $\mathcal{M}_0 = 2.7$.

$$\partial_x \left(\rho\, u\right) = 0\,, \tag{41a}$$

$$\partial_x \left(\rho\, u^2 + p + \frac{a_r\, \theta^4}{3}\right) = 0\,, \tag{41b}$$

$$\partial_x \left[u \left(\frac{1}{2}\rho\, u^2 + \rho\, e + p + \frac{4}{3\,\epsilon}\, a_r\, \theta^4\right)\right] = \frac{4\, a_r\, c}{3\,\epsilon}\partial_x \left(\frac{\theta^3}{\sigma_t}\partial_x \theta\right)\,, \tag{41c}$$

$$\rho\, u\, \partial_x e + p\, \partial_x u = \frac{\sigma_a\, c}{\epsilon}\left(\mathcal{E} - a_r\, T^4\right) - 2\,\sigma_a\, \frac{u}{c}\, \mathcal{F} + \frac{u^2}{c}\, \left(\sigma_t\, \mathcal{P} + \sigma_s\, \mathcal{E} + \sigma_a\, a_r\, T^4\right)\,. \tag{41d}$$

While it is not obvious from the equations above, the only effect of the scalings is to divide the radiation energy source, S_{re}, by ϵ. Thus, in the energy equation the radiation terms are divided by ϵ, and in the internal energy equation only the term coming from the radiation energy source is divided by ϵ. Subsequently, as $\epsilon \to 0$, the internal energy equation will push the solution toward equilibrium. These new results, and their solution method was documented in the open literature [6] as a part of this L2 Milestone effort.

E. The grey S_n-transport radiation model

As a reminder, the equilibrium-diffusion approximation assumes, in addition to Eulerian hydrodynamics, [26]

1) that the fluid-radiation system is in thermal equilibrium,
2) that the radiation pressure is isotropic, and
3) that the mean-free-path for interactions between the fluid and radiation in the system is much smaller than the system's size so that it may be reasonably assumed that the radiation diffuses through the system.

The nonequilibrium-diffusion approximation relaxed the first assumption, allowing the fluid to have a material temperature, T, the radiation to have a separate temperature, θ, and the radiation energy density to be defined with respect to the radiation temperature, $\mathcal{E} = a_r\, \theta^4$. The S_n-transport approximation uses a variable Eddington factor (VEF), f, to relate the radiation pressure to the radiation energy density, $\mathcal{P} = f\, \mathcal{E}$. How the VEF varies spatially is determined by solving the grey S_n-transport equation, to be discussed below. Since the shocktube is assumed to be "infinitely" long, we retain the assumption that the radiation mean-free-path in the fluid is much smaller than the size of the system, but we abandon the diffusion approximation. This is not a drastic change however, since the only change is that the radiation flux is now the negative gradient of the radiation pressure instead of the radiation energy density, $\mathcal{F} = -\left[c/(3\,\sigma_t)\right]\partial_x \mathcal{P} + u(\sigma_t\, \mathcal{P} + \sigma_s\, \mathcal{E} + \sigma_a\, a_r\, T^4)$. A fundamental assumption of the S_n-transport method is that the angular dependence of the radiation intensity can be reasonably represented by a finite number of directions, and angular integrals are reduced to quadrature. Reducing these integrals to quadrature sums introduces a weight and root along each direction, so that for N directions, with $m = 1 \ldots N$, there exists a weight, w_m, and root, μ_m, along each direction. Sets of these weights and roots containing specific mathematical features are well known, and go by the names of "Gauss-Legendre" or "Gauss-Lobatto" and the like, depending on one's needs. For the purposes of our solutions we use Gauss-Legendre weights and roots.

In the S_n-transport approximation the RH equations are:

$$\partial_x \left(\rho\, u\right) = 0\,, \tag{42a}$$

$$\partial_x \left(\rho\, u^2 + p\right) = -S_{rp}\,, \tag{42b}$$

$$\partial_x \left[u\left(\frac{1}{2}\rho\, u^2 + \rho\, e + p\right)\right] = -S_{re}\,, \tag{42c}$$

$$\mu_m\, \partial_x I_m\left(\mu_m\right) = Q_m\left(\mu_m\right)\,, \tag{42d}$$

$$Q_m\left(\mu_m\right) = -\sigma_t\, I_m\left(\mu_m\right) + \sigma_a \frac{a_r\, c\, T^4}{4\pi} + \frac{\sigma_s}{4\pi}\mathcal{E}\,, \tag{42e}$$

$$S_{re} = \sum_{m=1}^{N} w_m\, Q_m\left(\Omega_m\right)\,, \tag{42f}$$

$$S_{rp,i} = \frac{1}{c}\sum_{m=1}^{N} w_m\, \Omega_m\, Q_m\left(\Omega_m\right)\,, \tag{42g}$$

$$\mathcal{E} = \sum_{m=1}^{N} w_m\, I_m\left(\Omega_m\right)\,, \tag{42h}$$

$$\mathcal{F} = \sum_{m=1}^{N} w_m\, \mu_m\, I_m\left(\Omega_m\right)\,, \tag{42i}$$

$$\mathcal{P} = \sum_{m=1}^{N} w_m\, \mu_m^2\, I_m\left(\Omega_m\right)\,. \tag{42j}$$

Reviewing equations (37), we see that the S_n-transport RH equations are quite similar except for the integrals which have been reduced to quadrature. In addition to the RH equations we still need the internal energy equation in order to have a solvable system:

$$\rho\, u\, \partial_x e + p\, \partial_x u = -S_{rie} = \sigma_a\, c\, \left(\mathcal{E} - a_r\, T^4\right) - 2\,\sigma_a\, \frac{u}{c}\, \mathcal{F} + \frac{u^2}{c}\, \left(\sigma_t\, \mathcal{P} + \sigma_s\, \mathcal{E} + \sigma_a\, a_r\, T^4\right)\,. \tag{42k}$$

To solve this system of equations we begin with a nonequilibrium-diffusion radiative-shock solution, obtained from the conservation statements for mass, total momentum and total energy, as well as the internal energy equation. Then the S_n transport equation (42d and 42e) is solved for the radiation intensity, I_m, along each quadrature direction, $m = 1 \ldots N$, while assuming that all other parameters are known from the solution of the conservation statements and the internal energy equations. The radiation energy density and radiation pressure may then be re-constructed and their ratio used to determine the VEF, $f = \mathcal{P}/\mathcal{E}$. Then the conservation statements and the internal energy equation are solved again, but now the VEF is used as necessary for the radiation energy density. While approximations described in the subsections above retained the radiation energy density as an independent variable, that role has been given to the radiation pressure in our solutions described in this subsection. This is partly because the radiation pressure occurs more often than the radiation energy density, but mainly because the gradient for the radiation flux depends on the radiation pressure.

Equations (42) support radiative-shock solutions as shown in Figure (83). Now, however, we also obtain information about the angular-dependence of the radiation field. Further, a fundamental, and often unspoken, result of the diffusion approximation described in previous subsections is that the radiation energy is monotonic. As shown in Figure (83), the S_n-transport solutions support nonmonotonic radiation energy solutions. These new results and others, along with their solution method were documented in the open literature [5] as a part of this L2 Milestone effort.

F. The ion-electron shock problem

The solution method for the ion-electron shock solution [23] is similar in structure to the nonequilibrium-diffusion solution method, although the equations are obviously different. The model developed in that paper begins with the mass and momentum conservation statements from Eulerian hydrodynamics, couples electron thermal diffusion to the energy conservation statement, and solves the electron internal energy equation. Ion mobility or viscosity are neglected. As such, there are separate ion and electron temperatures which can be combined into an material temperature. In the limit when the ion and electron temperatures

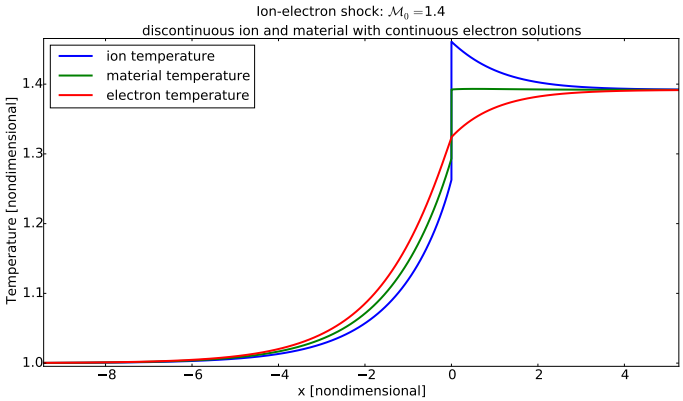


Fig. 84: The ion-electron shock solution for $\mathcal{M}_0 = 1.4$.

are constrained to be equal then the model reduces to that of Eulerian hydrodynamics with material heat conduction. The model equations are:

$$\partial_x \left(\rho\, u\right) = 0\,, \tag{43a}$$

$$\partial_x \left(\rho\, u^2 + p\right) = 0\,, \tag{43b}$$

$$\partial_x \left[u\left(\frac{1}{2}\rho\, u^2 + \rho\, e + p\right)\right] = \partial_x \left(\kappa_e\, \partial_x T_e\right)\,, \tag{43c}$$

$$\rho\, u\, \partial_x e_e + p_e\, \partial_x u = \gamma_{ei}\left(T_i - T_e\right) + \partial_x \left(\kappa_e\, \partial_x T_e\right)\,. \tag{43d}$$

An ion-electron shock solution from these equations is presented in Figure 84. The ion temperature, material temperature and electron temperature are presented there. An embedded hydrodynamic shock is present in the ion and material temperatures. However, it is an assumption of the solution method that the electron temperature is continuous, therefore it is only discontinuous in its derivative at the embedded hydrodynamic shock. A xRAGE input deck has not been built for this problem, so there are no xRAGE results, yet.

VI. Verification of Material Models

A. Summary

Two problems testing material models were implemented in ExactPack as part of extending the Verification Test Suite (VTS): the Blake problem [28] and the Hunter problem [29]. These two problems have historically been chosen to verify hydrocodes (for example, see [30]) because they are two of the few dynamic strength-of-materials problems that have closed-form solutions.

This report gives a high-level overview of the problems along with FLAG results compared to the exact solutions. FLAG performs well on both Blake and Hunter in terms of matching the primary features and converging to the exact solutions under mesh refinement. Consistent with the Sage results from [30] for the same problem parameters, FLAG matches the Blake problem exact solution well with a convergence order parameter of about 1/2. Restricting the fit to the smooth part of the solution results in an order parameter that rounds to one.

The Enhanced Tri-Lab Test Suite (eTLTS) for verification of physics simulation codes [18] gives parameters for using Hunter as a test problem; however, reports of using those parameters to verify hydrocodes at LANL indicate discrepancies between the hydrocode results and the exact solution [30], [31]. The exact solution itself is inconsistent with the parameters given because the solution assumes small strain while the results demonstrate substantial strain in the plastically deformed region. Thus, by reducing the yield strength, which also reduces the applied pressure and in turn the displacement, the exact solution is then being used within the derived regime. FLAG results then match the exact solution with a convergence order that varies depending on mesh resolutions used from about 2/3 for coarse meshes to substantially less for the last two finest meshes.

B. The Blake Problem

1) Overview of the Blake Problem: The Blake problem is a spherical symmetric problem that consists of a uniform, infinite, linear elastic material that has a spherical cavity of radius a centered at the origin. The closed-form of the solution involves applying a constant pressure to the inner cavity boundary so that it expands, sending a pressure wave through the material.

The derivations of the equations can be seen in [1]. The parameters used to set up this instance of the problem (see Table IV) come from [30]. In that paper, Kamm compared results from SAGE to the exact solutions. Key findings from [30] were a) the largest errors were near the cavity wall and at the elastic wave front and b) the convergence orders for quantities considered were closer to 0.5 than the expected first order for problems with a discontinuity.

Parameter	**Value**
Inner radius	a = 10 cm
Density	ρ = 3.0 g/cm^3
Bulk modulus	$K = 4.16\overline{6} * 10^{11}$ dyn/cm^2
Poisson ratio	ν = 0.25
Lamé constant	$\lambda = 2.50 * 10^{11}$ dyn/cm^2
Shear modulus	$\mu = 2.50 * 10^{11}$ dyn/cm^2
Long. wave speed	$c_L = 5.0 * 10^5$ cm/s
Shear wave speed	$c_S = 6.1237 * 10^5$ cm/s
Driving pressure	$P_0 = 10^7$ dyn/cm^2
Final time	$t_{fin} = 1.6 * 10^{-4}$ s
Max. mesh radius	r_{max} = 100 cm

TABLE IV: Parameters for the Blake Problem [30].

2) Blake Results and Analysis: The input decks for FLAG were set up according to the parameter table. The timestep in FLAG was allowed to adjust, but started at a small percentage of the CFL timestep. 10 mesh resolutions were run with one coarse mesh to show improvement and several fine meshes to investigate convergence order behavior upon significant mesh refinement.

Figures 85 through 94 show the pressure, radial stress deviator, hoop stress deviator, the absolute value of the stress difference, and the von Mises stress along with convergence plots of their L_1 error norms. As expected, the coarsest mesh resolution has the largest error in all cases. Refining the mesh leads to better agreement with smaller errors.

Restricting the error norm calculation regime to exclude the discontinuity (95) indicates that the convergence order is one. Restricting the error norm calculation regime to have only the discontinuity (96) indicates a convergence order of 0.479. Thus, the discontinuity itself is causing most of the deviation from first order convergence.

The initial peak/valley in the stress deviators have been clipped in the figures so the elastic wave front can be clearly seen. FLAG smooths out the discontinuity at the elastic wave front for coarser grids. The absolute value of the stress differences shows a bump (a reversal of which pressure is largest) before the spike associated with the elastic wave front. This correlates with the feature in the pressure plot in the same region.

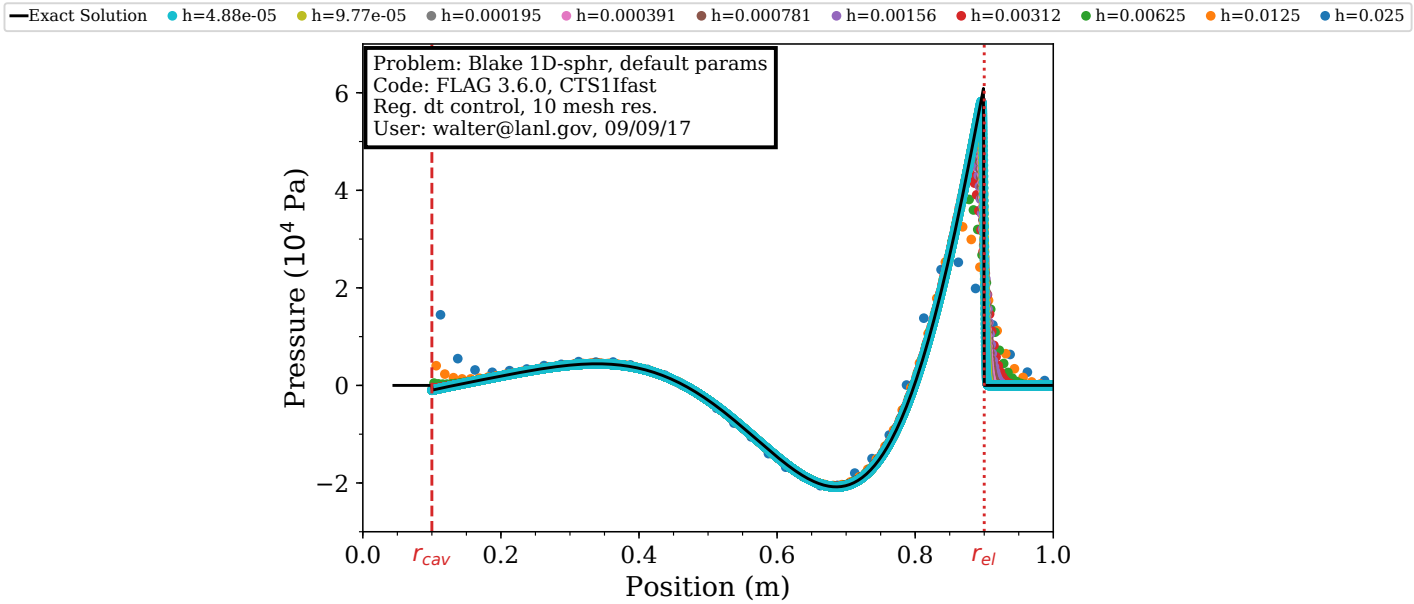


Fig. 85: Pressure results from FLAG for Blake's problem. Resolutions in m.

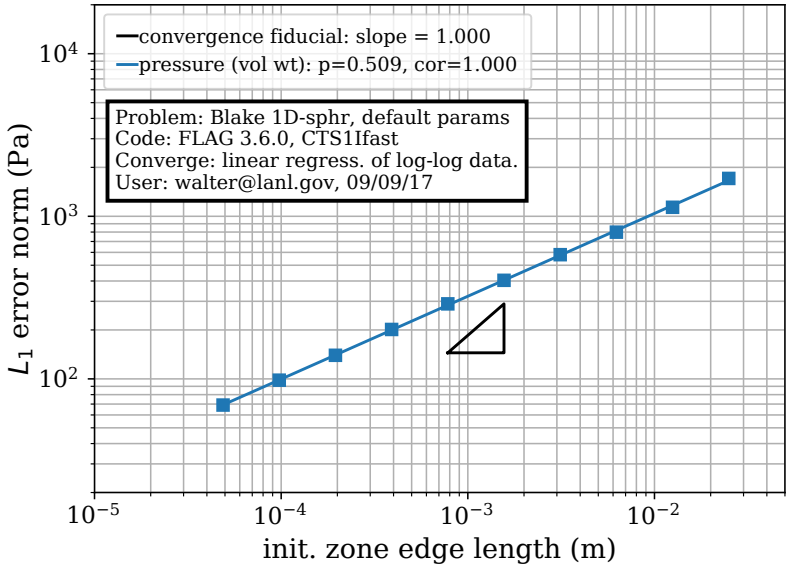


Fig. 86: Pressure L_1 error norm convergence results from FLAG for Blake's problem.

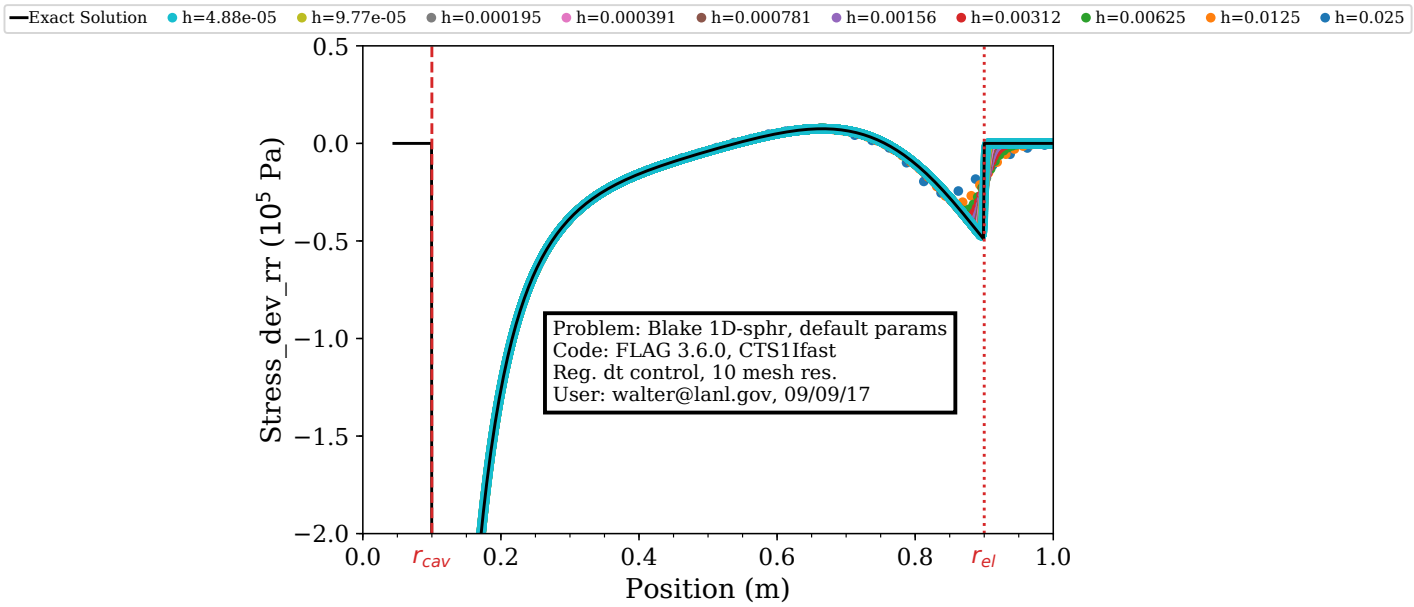


Fig. 87: Radial stress deviator results from FLAG for Blake's problem. Resolutions in m.

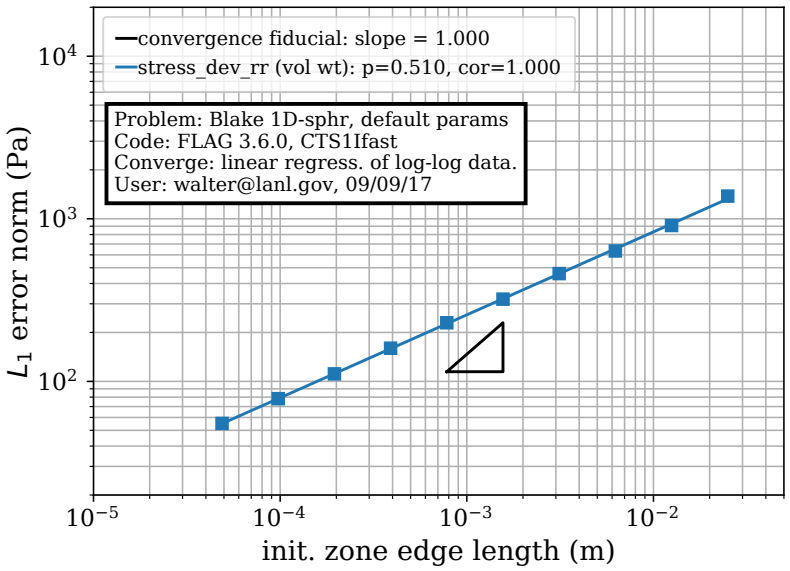


Fig. 88: Radial stress deviator L_1 error norm convergence results from FLAG for Blake's problem.

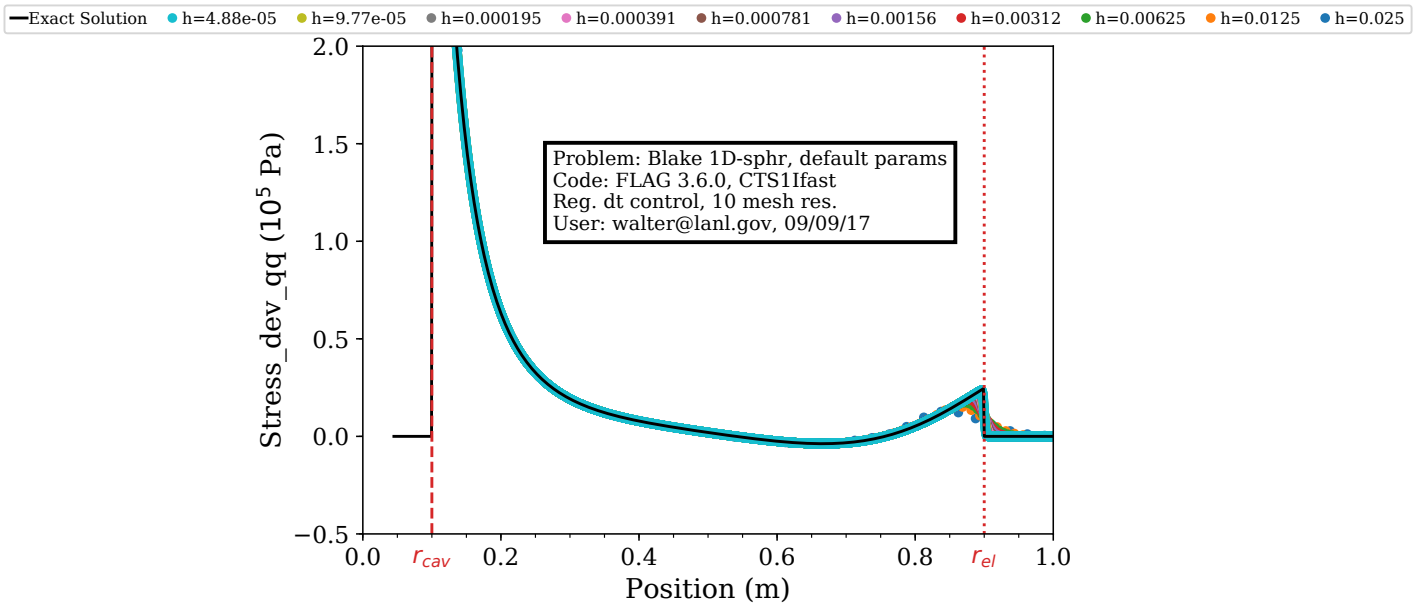


Fig. 89: Hoop stress deviator results from FLAG for Blake's problem. Resolutions in m.

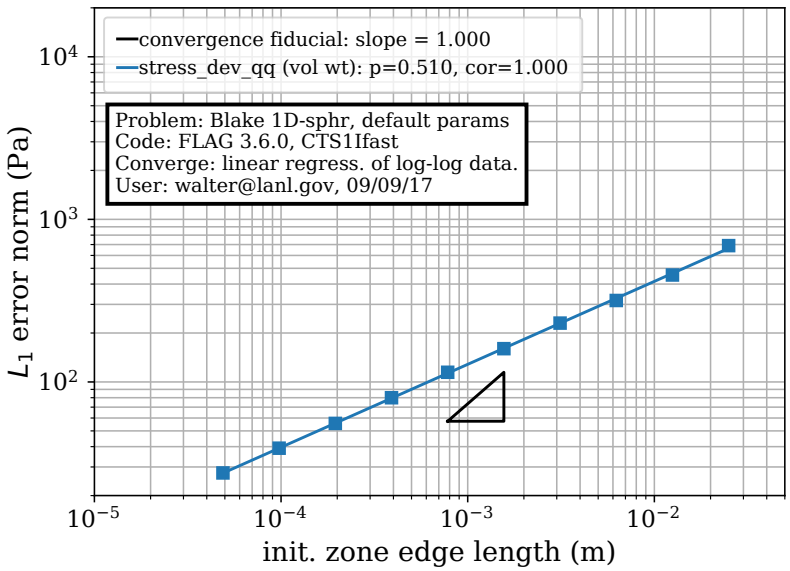


Fig. 90: Hoop stress deviator L_1 error norm convergence results from FLAG for Blake's problem.

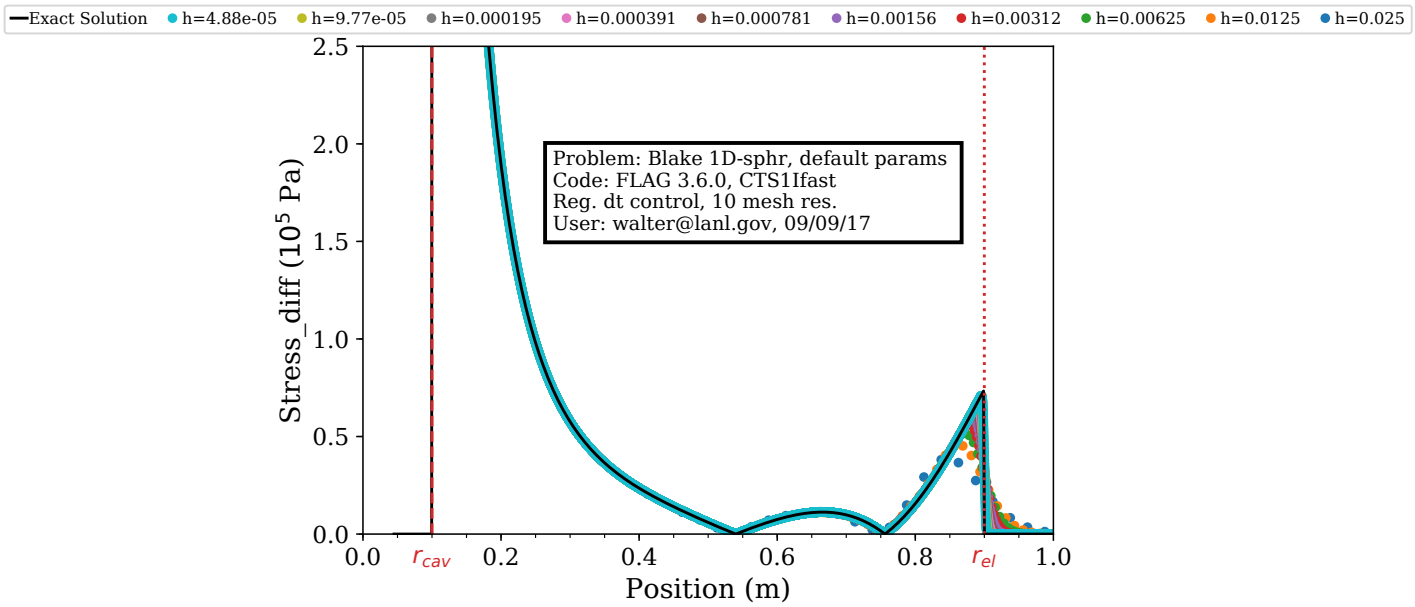


Fig. 91: Absolute value of stress difference results from FLAG for Blake's problem. Resolutions in m.

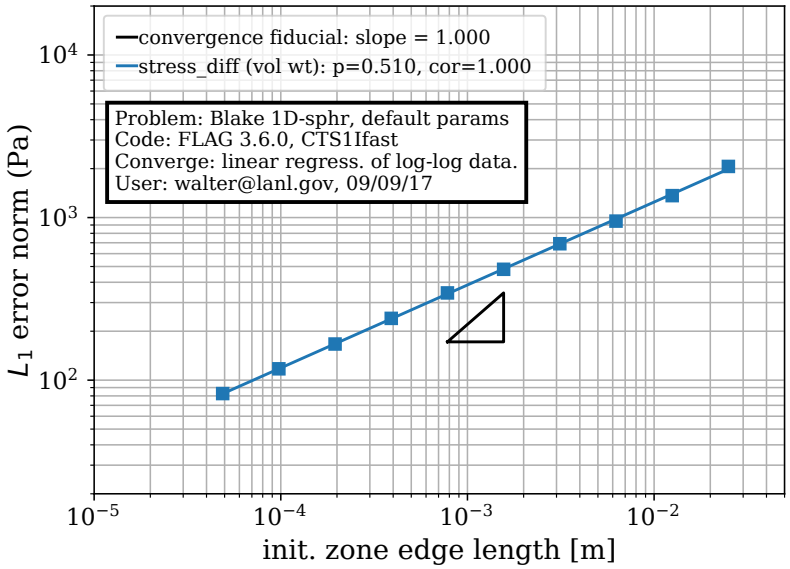


Fig. 92: Absolute value of stress difference L_1 error norm convergence results from FLAG for Blake's problem.

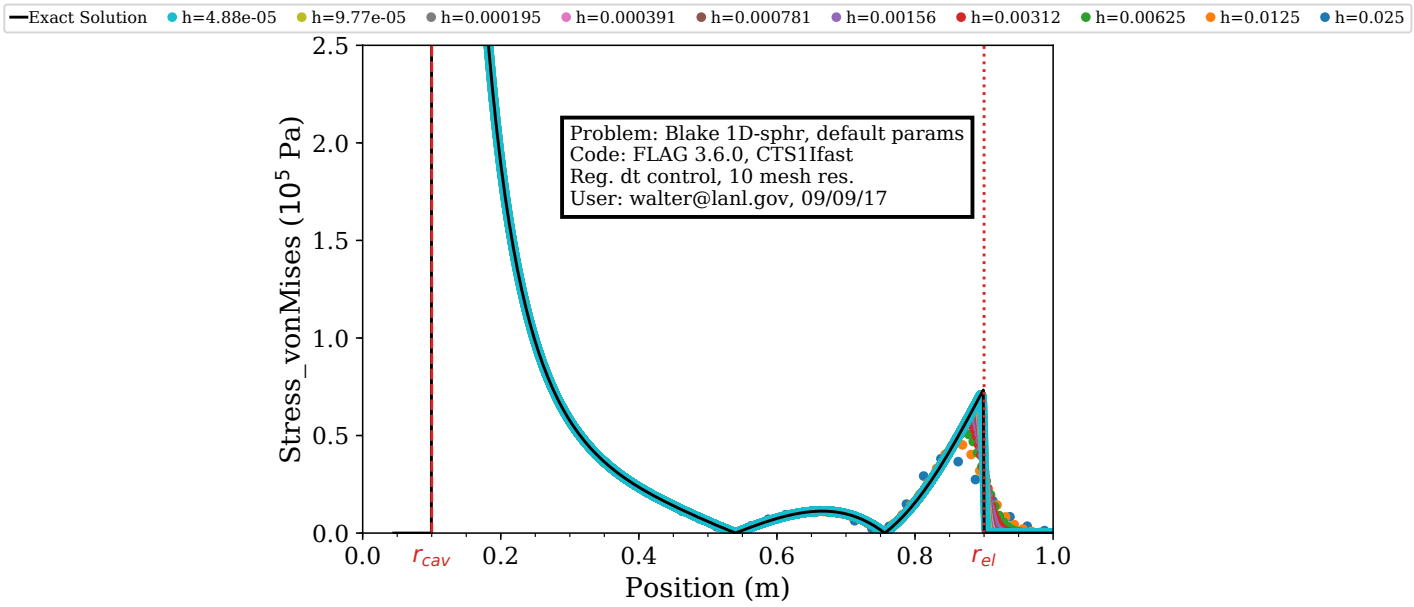


Fig. 93: von Mises stress results from FLAG for Blake's problem. Resolutions in m.

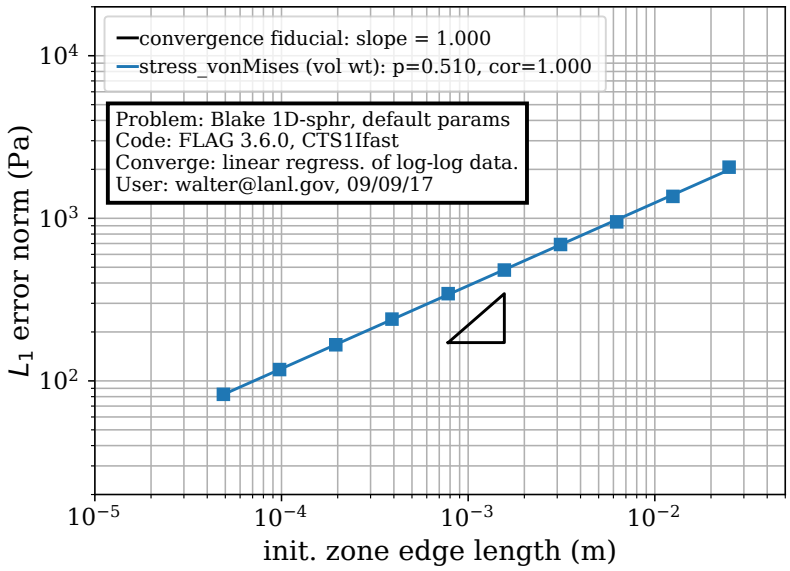


Fig. 94: von Mises stress L_1 error norm convergence results from FLAG for Blake's problem.

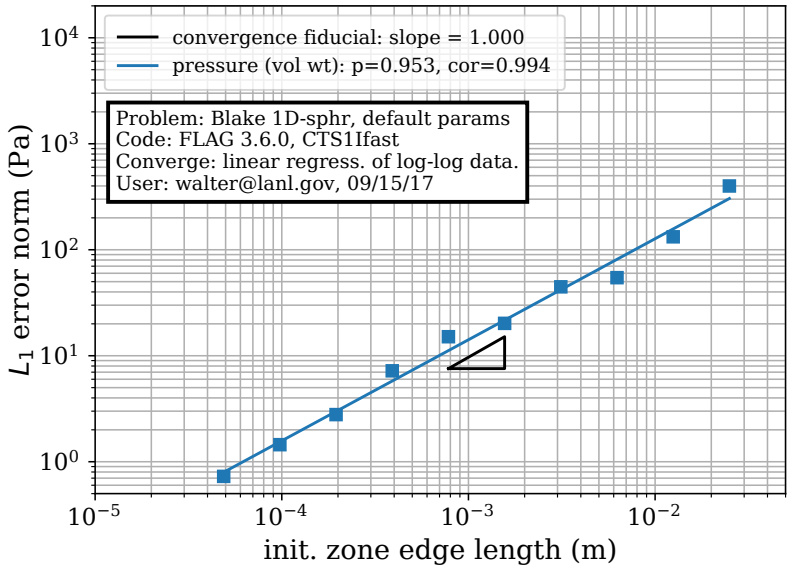


Fig. 95: Pressure L_1 error norm convergence results from FLAG for Blake's problem restricted to 0.15 m to 0.85 m.

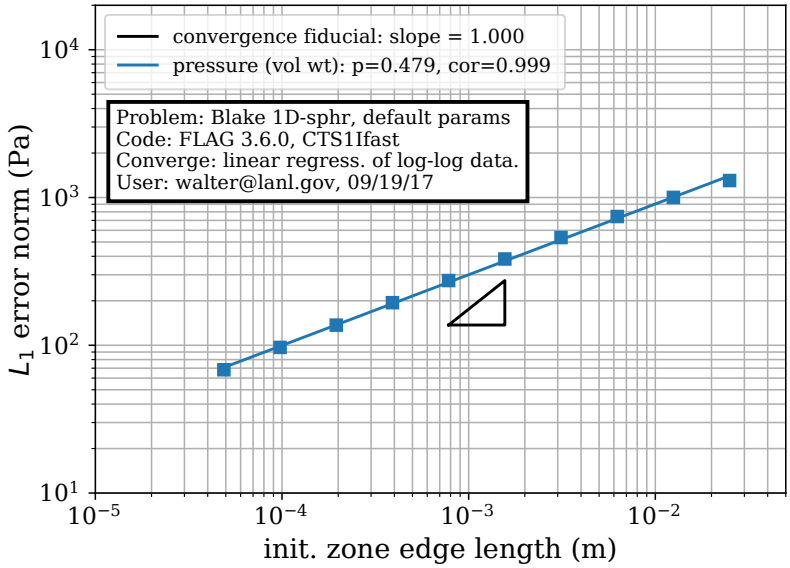


Fig. 96: Pressure L_1 error norm convergence results from FLAG for Blake's problem restricted to 0.85 m to 1.0 m.

C. The Hunter Problem

1) Overview of the Derivation of the Hunter Problem: Hunter [29] solved the problem of an elastic-perfectly plastic material that has an expanding cavity producing small-amplitude waves that deform the material.

The underlying method is solving the wave equation in three regions:

1) plastically deformed region ($r_{cav} \leq r < r_{ep}$)
2) elastically deformed region ($r_{ep} \leq r \leq r_{el}$)
3) undisturbed region ($r_{el} < r$)

The solutions are then used to calculate quantities of interest including displacement, stress, and strain.

Derivation of the equations are given in [29]. Key assumptions in the derivations include:

1) linear elastic, perfectly plastic, uniform, infinite material.
2) small strain response.
3) a constant velocity for the elastic-plastic boundary (c_{ep}). By assuming the position of that boundary has the definite form

$$r_{ep} = r_{cav} + c_{ep}t, \tag{44}$$

the wave equations can be solved analytically with the resulting code having only algebraic expressions.

The pressure applied to the cavity boundary is calculated such that the input value of c_{ep} will be obtained and remain constant as a function of time. Code for calculating that pressure and outputting it in a format suitable for FLAG is included within the Hunter solver in ExactPack. The complexity of the calculation is such that performing the calculation from within a FLAG input deck was not attempted for this work.

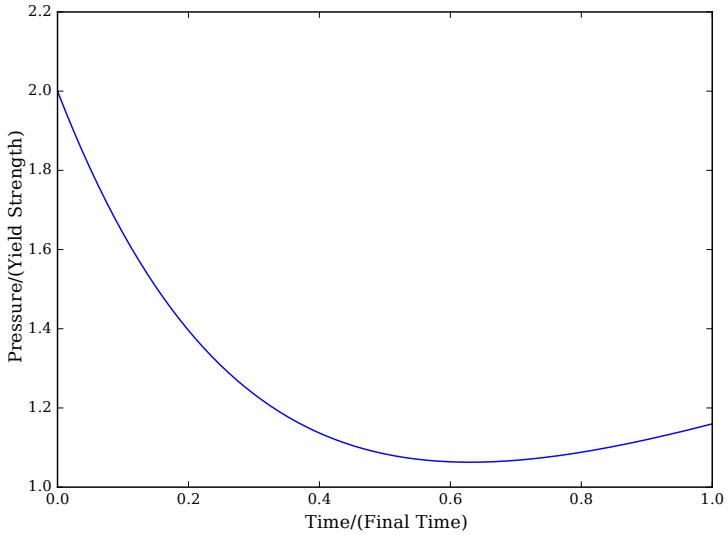


Fig. 97: Applied pressure on the inner boundary in non-dimensional units.

Typographical errors are corrected here to aid the reader consulting the original paper [29] for derivation. Those errors are:

- All log notations are natural logarithms.
- In (4.39), the denominator of the final term should be $c_1^2 r$.
- In (4.67), all denominators should be $n_1 - n_2$. (4.78) is the same equation and thus should have n_2 in the second exponent. Both equations should appear as:

$$g_1(x) = h\left\{\frac{3-n_2}{n_1-n_2}x^{n_1} - \frac{3-n_1}{n_1-n_2}x^{n_2} - x^3\right\} \tag{45}$$

- In (4.77), the first $3 - n_1$ term should be $3 - n_2$, just as in (4.76).

2) Review of Previous Work: The Hunter problem is part of the proposed Enhanced Tri-Lab Test Suite (eTLTS) [18] because it is one of the few closed-form solutions available for dynamic material response. The Hunter problem is a generalization of the Blake problem.

Parameter	**Value**
Cavity boundary	a = 100 cm
Density	ρ = 1.0 g/cm^3
Bulk modulus	K = 10^{10} dyn/cm^2
Poisson ratio	ν = 1/3
Lamé constant	$\lambda = 7.5 * 10^9$ dyn/cm^2
Shear modulus	$\mu = 3.75 * 10^9$ dyn/cm^2
Long. wave speed	$c_L = 1.2247 * 10^5$ cm/s
Shear wave speed	$c_S = 6.1237 * 10^4$ cm/s
Plastic wave speed	$c_P = 10^5$ cm/s
EP interface speed	$c_{EP} = 2.4495 * 10^4$ cm/s
Yield strength	$Y = 10^9$ dyn/cm^2
Driving pressure	evaluated by exact solution code
Final time	t_{fin} = 0.001 s
Max. mesh radius	r_{max} = 300 cm

TABLE V: Enhanced Tri-Lab Test Suite Parameters for Hunter's Problem [18].

Kamm performed validation of the LANL code SAGE using Hunter and the eTLTS parameters. Kamm found that "the displacement of the stresswave-processed material is appreciable at the final time" [30]. This is true from the exact solution using the parameters in Table V (see Fig. 98).

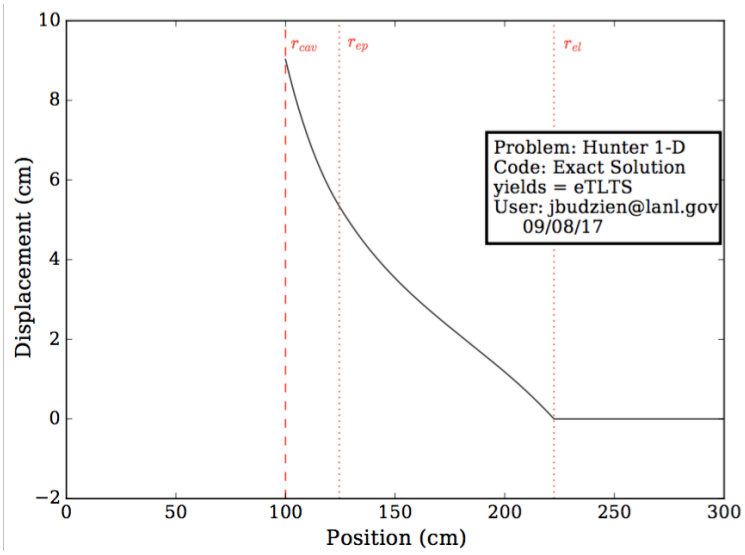


Fig. 98: Displacement for the exact solution using the eTLTS parameters. Note that the cavity boundary moves about 9 cm from its original position of 100 cm from the origin.

The exact solution is formulated relative to the initial positions while hydrocodes often report at current position. Naively making that comparison with the FLAG results gives Fig. 99. Performing a transformation from initial positions to current positions, as Kamm did for the SAGE comparison, does resolve the discrepancy in feature position, but does not revolve the difference in magnitude of the quantity in question, particularly at the elastic-plastic boundary.

While the solutions are clearly converging to an answer under mesh refinement, Kamm notes approximately zeroth-order convergence of the L_1 error norm for the Hunter problem in SAGE [30] and that is true in FLAG as well (see Fig. 100). Kamm hypothesized about possible reasons for the discrepancy. On page 17 of Reference [30], the first point is "Errors in the exact solution ... (v) the apparent violation of the strict small-strain assumption for the parameters considered". The yield strength as listed in the eTLTS parameter set is a much larger fraction of the bulk modulus than is typical for common metals such as alumimum and copper. Reducing the yield strength will allow for yield under much less pressure, which will also be much less displacement for both the plastic and elastic regions to be more in accord with the assumption of small strain. The

yield strength appears in the equations as a prefactor for the most part. Hunter reports all quantities normalized in some way by yield strength; even strain is reported in units of yield strength divided by twice the shear modulus.

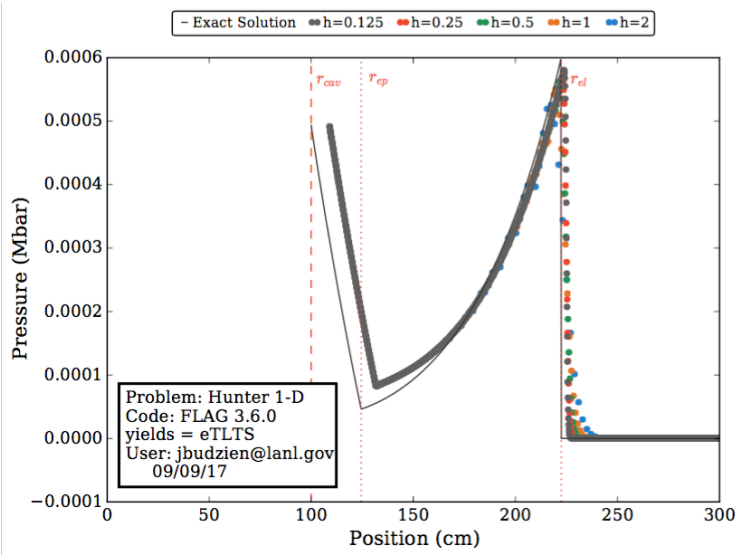


Fig. 99: Pressure results for Hunter using FLAG and the eTLTS parameter set.

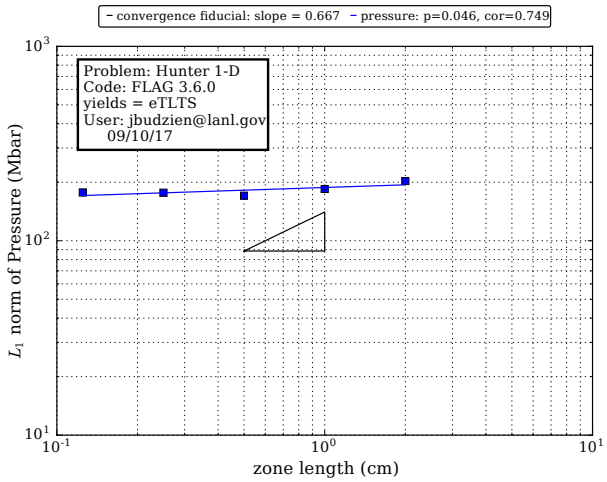


Fig. 100: Convergence of the L_1 error norm for pressure using the eTLTS parameters.

Consequently, the yield strength was reduced to one-tenth (0.1) of its original value, while keeping all other parameters the same, and the FLAG simulations were run again. The displacement is now within a small strain limit for all regions of the material (see Fig. 101).

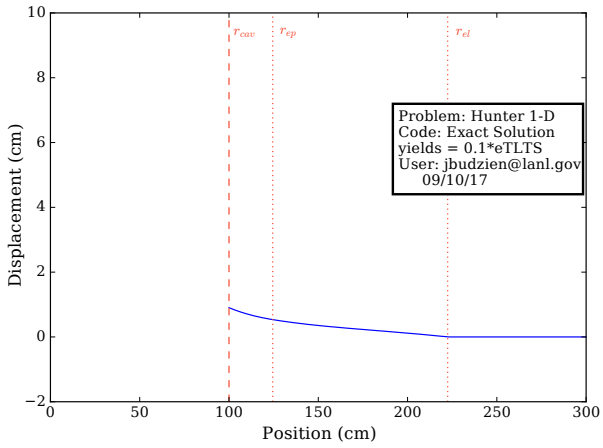


Fig. 101: Exact solution displacement for Hunter's problem using a yield strength 0.1*(yield strength from eTLTS).

3) Hunter Results and Analysis: Using the parameters from the eTLTS with the smaller yield strength has resulted in FLAG performing very well compared to the exact Hunter solution. Applying the full pressure at time 0 required an initially small timestep that was allowed to grow to the CFL timestep for the smallest mesh resolution. More mesh resolutions were used than required by the eTLTS problem requirements to have much finer resolutions as well as one coarser resolution (50 zones, 4 cm) that shows significant error.

Figures 102 through 110 show the pressure, radial stress deviator, hoop stress deviator, and the von Mises stress along with convergence plots of their L_1 error norms. As expected, the coarsest mesh resolution has the largest error in all cases. Refining the mesh leads to better agreement with smaller errors.

FLAG captures the plastic deformation region extremely well, even for the coarsest resolution. The elastic-plastic boundary is also captured well. The largest error in every quantity is near the elastic wave front (r_{el} on the figures), much like Blake. FLAG smooths out the sharp discontinuity, although finer meshes are significantly better at performing less smoothing as one might hope.

The convergence order of the L_1 error norm plots are consistent across quantities for the same mesh resolutions, but are not linear on the log-log plots. Instead, the convergence order clearly changes based on which subset of mesh resolutions are chosen. The coarsest grids yield convergence orders that might be considered 2/3. However, the finest meshes show much lower convergence orders and may be approaching a number close to zero.

Reducing the yield strength by another factor of 10 (e.g., (yield strength)/100), while keeping all other parameters the same, reduces the L_1 error norm magnitude as expected due to the smaller values of all quantities (see Fig. 104). However, that reduction also shows a more linear profile for the convergence plot of the L_1 error norm even at fine meshes. The convergence order for the finest meshes appears to be 0.5, much like the Blake problem.

Only the pressure convergence plot is shown for the set of parameters that include (yield strength)/100. No additional information is provided by showing these figures as well because the exact solution is scaled by yield strength, which results in the same overall shape with only a change in scale.

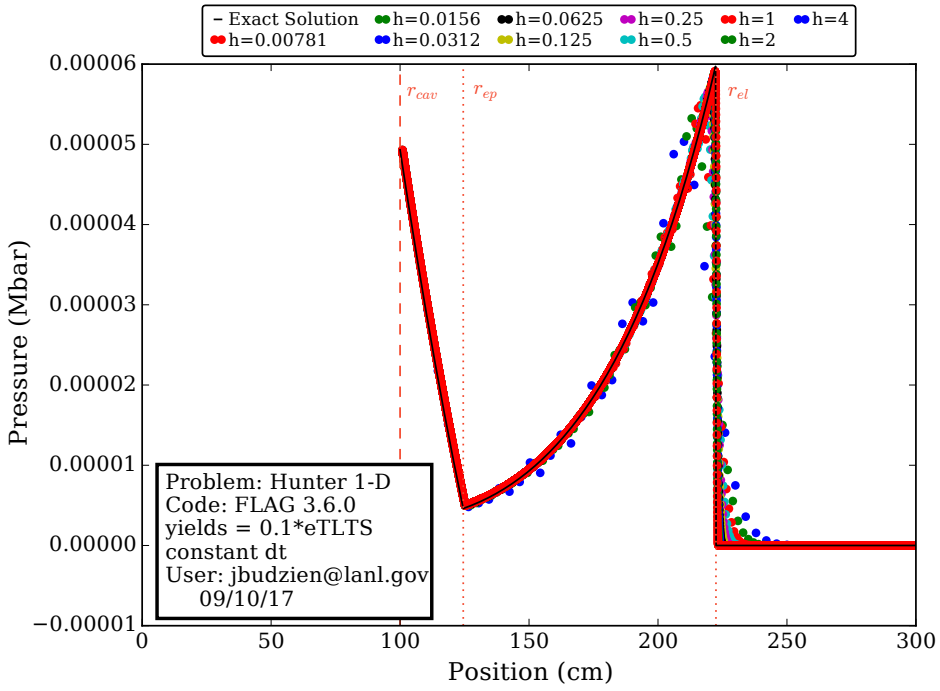


Fig. 102: Pressure results from FLAG for Hunter's problem with the modified eTLTS parameters. Resolutions in cm.

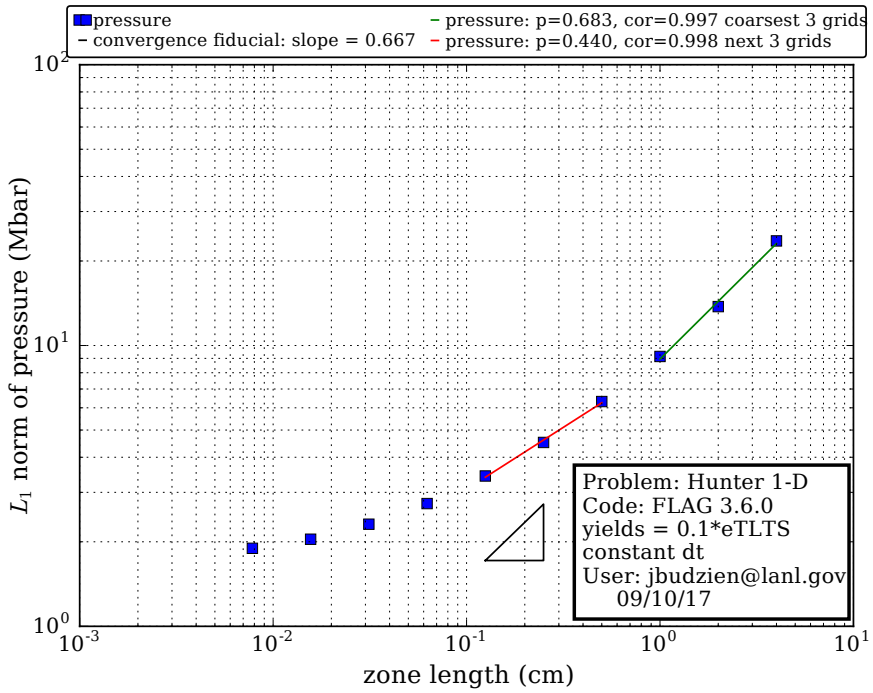


Fig. 103: Pressure L_1 error norm convergence results from FLAG for Hunter's problem with the modified eTLTS parameters.

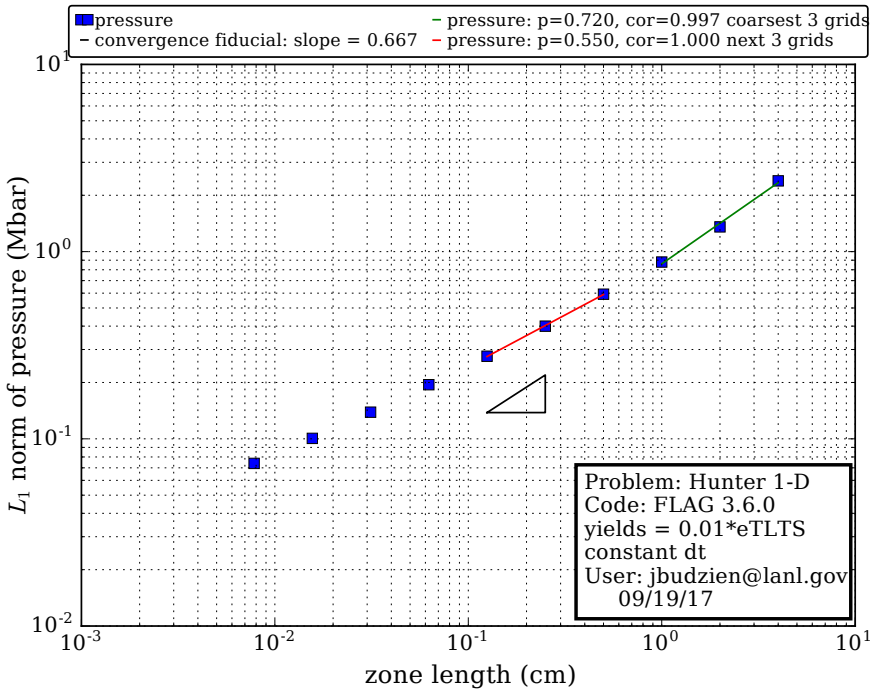


Fig. 104: Pressure L_1 error norm convergence results from FLAG for Hunter's problem with the (eTLTS yield strength)$/100$.

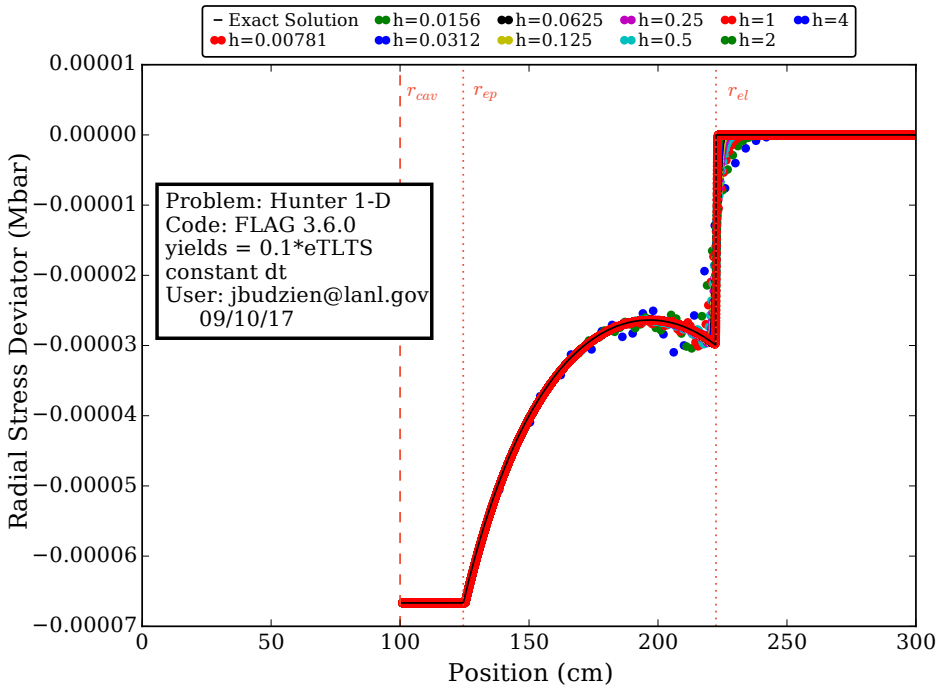


Fig. 105: Radial stress deviator results from FLAG for Hunter's problem with the modified eTLTS parameters. Resolutions in cm.

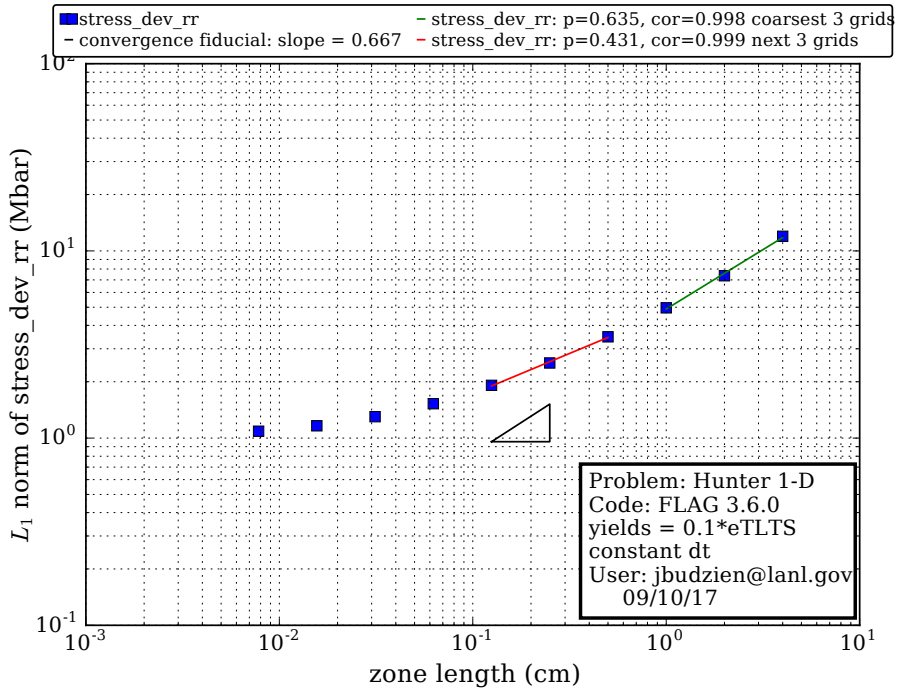


Fig. 106: Radial stress deviator L_1 error norm convergence results from FLAG for Hunter's problem with the modified eTLTS parameters.

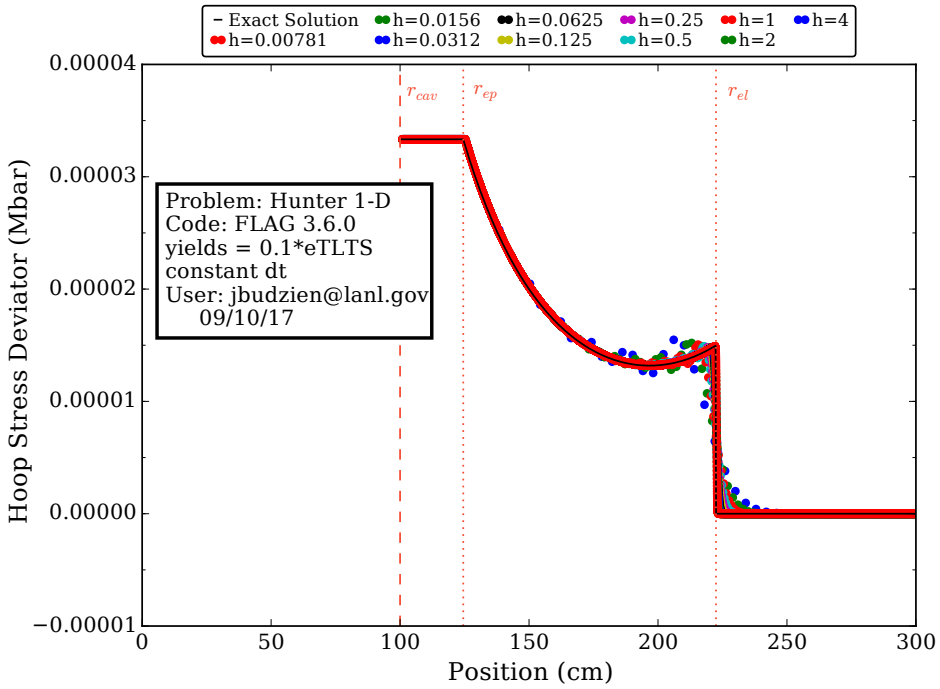


Fig. 107: Hoop stress deviator results from FLAG for Hunter's problem with the modified eTLTS parameters. Resolutions in cm.

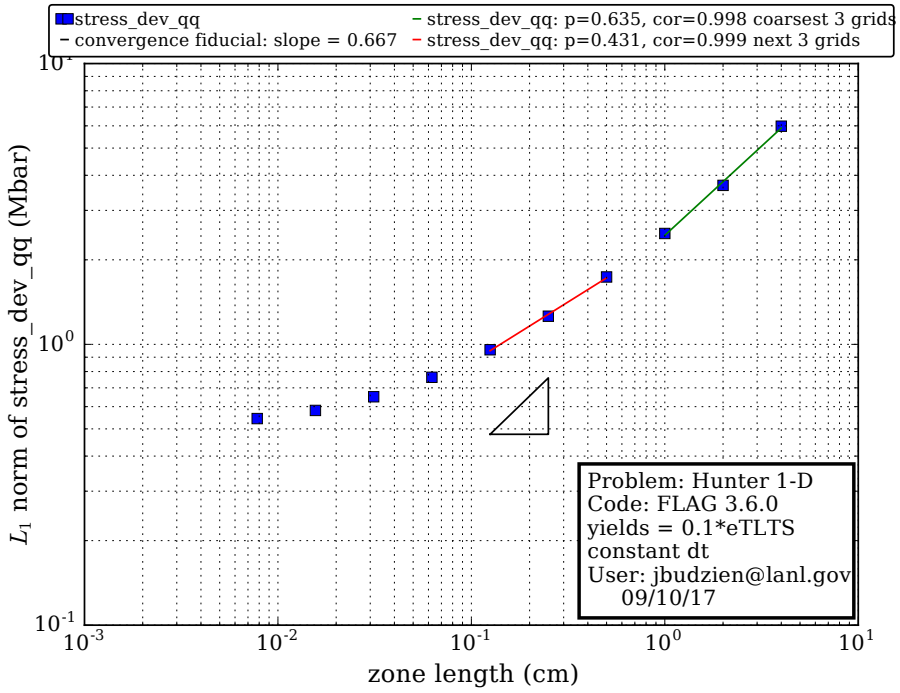


Fig. 108: Hoop stress deviator L_1 error norm convergence results from FLAG for Hunter's problem with the modified eTLTS parameters.

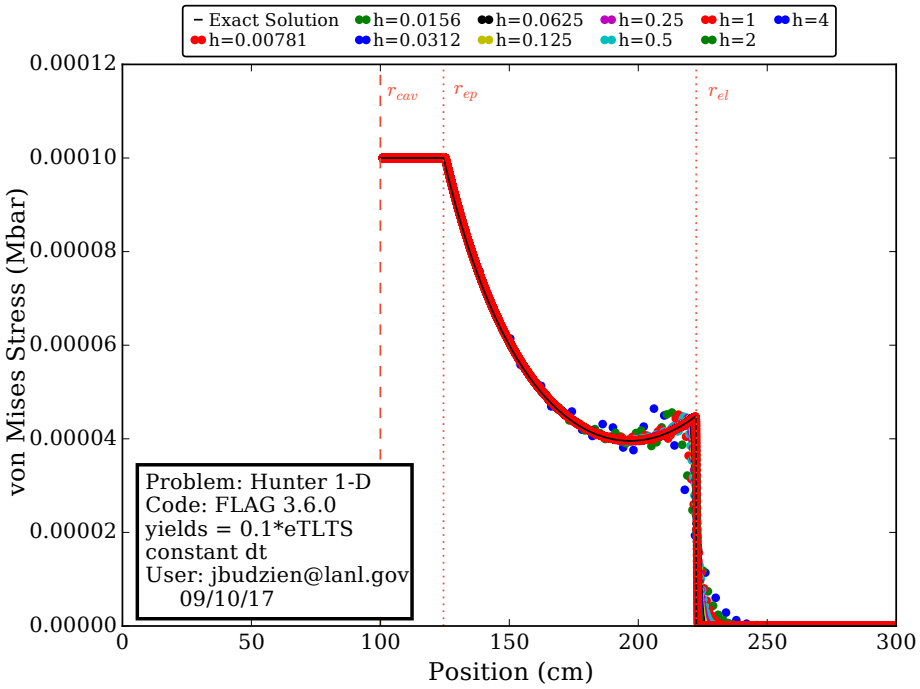


Fig. 109: von Mises stress results from FLAG for Hunter's problem with the modified eTLTS parameters. Resolutions in cm.

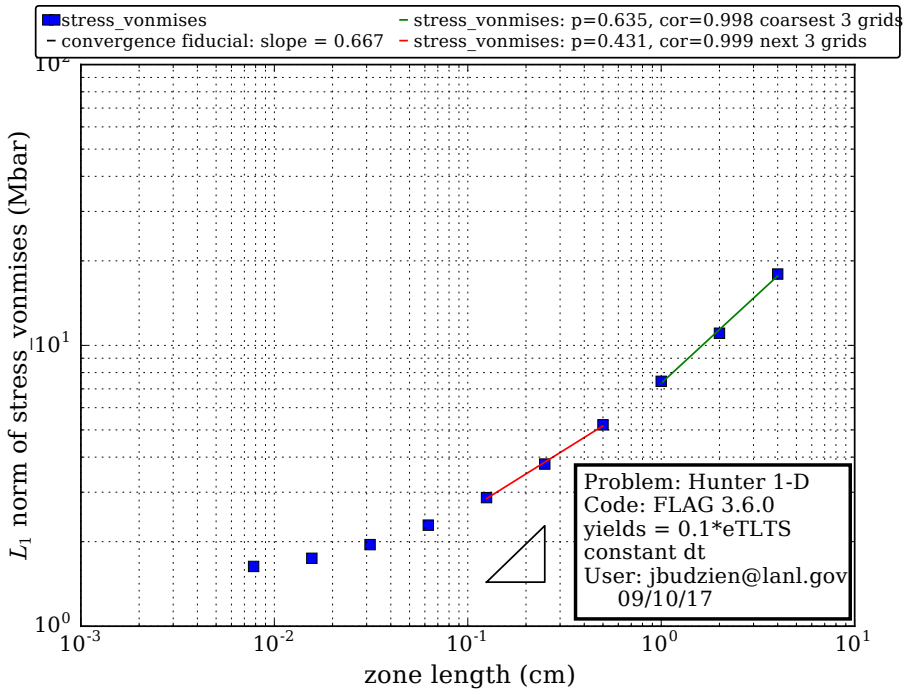


Fig. 110: von Mises stress L_1 error norm convergence results from FLAG for Hunter's problem with the modified eTLTS parameters.

VII. Verification of Heat Flow

We perform a code verification convergence analysis for FLAG using a newly constructed 2D heat flow test problem called the *planar sandwich* [32]. This solution has been analyzed by Dawes, Shashkov, and Malone [33] in the context of multimaterial diffusion. By a *multimaterial* cell, we mean a computational cell containing multiple materials, each with their own distinct physical properties. In the context of heat flow, a multimaterial cell contains a number of individual materials labeled by index m and heat conductivity κ_m. Subgrid models must be employed to resolve such physics in a hydrocode. The simplest approach is to replace a multimaterial cell by a uniform *average* material with a single conductivity $\bar{\kappa}$, where the average material is meant to reproduce the collective effects of the individual sub-materials with different values of κ_m. It is not clear that one can always do this faithfully, particularly when the material boundaries are misaligned with the computational grid. See Fig. 111 for a concrete illustration of a multimaterial cell and its corresponding numerical grid. FLAG has a number of algorithms for dealing with multimaterial cells:

1. Arithmetic average
2. Harmonic average
3. Thin mesh
4. Static condensation

1) Summary of Observations: The purpose of this section is to assess the relative accuracy and convergence rates of these various algorithms using the planar sandwich test problem. Figs. (116) and (117) show that the arithmetic and harmonic averages are 1^{st} order convergent, while Figs. (119) and (**??**) show that thin mesh and static condensation start out 2^{nd} order convergent (and then decrease in convergence rate as the method is iterated and the mesh is refined).

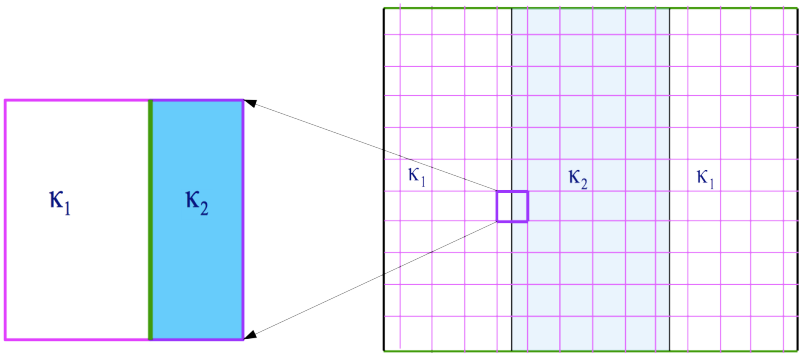


Fig. 111: The *planar sandwich* on a square numerical grid. The right most figure shows a numerical grid overlaid upon a rectangular physical geometry consisting of three parallel regions. The numerical grid partions the physical geometry into a number of corresponding numerical regions, none of which need be aligned with the physical regions. In the planar sandwich, the outer two regions are composed of the same material with diffusion coefficient κ_1, and the inner region is placed symmetrically about the origin of the x-axis and has diffusion coefficient κ_2. Note that the numerical grid is misaligned relative to a material boundaries of the middle region, and therefore multimaterial cells exist along the material boundary, as illustrated in left most figure.

2) The Numerical Issues: Let us examine the arithmetic and harmonic averaging techniques in more detail. As pointed out above, hydrocodes require that each individual cell contains a single uniform material, and the simplest method of handling multimaterial cells is to construct a uniform *average* material from the multiple materials. One can assign a single heat coefficient $\bar{\kappa}$ to a multimaterial cell by performing an averaging technique over the nonhomogeneous components of the cell, such as a (volume weighted) arithmetic or harmonic mean, $\bar{\kappa}_a$ or $\bar{\kappa}_h$, defined by

$$\bar{\kappa}_a = \sum_m V_m \, \kappa_m \tag{46}$$

$$\bar{\kappa}_h^{-1} = \sum_m V_m \, \kappa_m^{-1} \ , \tag{47}$$

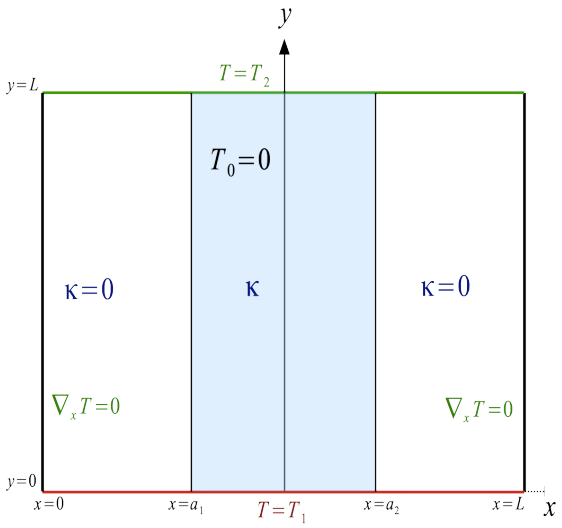


Fig. 112: The Planar Sandwich. The inner material in blue (the meat) is a heat conducting material with $\kappa > 0$, and is located within $a_1 \leq x \leq a_2$. The outer materials (the bread) are located within $0 \leq x < a_1$ or $a_2 < x \leq L$, are not heat conducting, so that $\kappa = 0$ in these regions. In numerical work, we take $\kappa = 10^{-12}$ rather than zero. The boundary temperature is uniform in x along the lower and upper boundaries, with Dirichlet boundary conditions $T(x, 0) = T_1$ and $T(x, L) = T_2$. When $T_1 > T_2$, heat flows uniformly upward from $y = 0$ to $y = L$. The temperature flux along the far left and right boundaries vanishes, $\partial_x T(x = 0, y) = 0$ and $\partial_x T(x = L, y) = 0$. Finally, the initial temperature vanishes inside the sandwich, $T_0(x, y) = 0$ for $(x, y) \in (0, L) \times (0, L)$. For numerical work, we take $a_1 = 0.75$ and $a_2 = 1.25$, or a slightly shifted variant with $a_1 = 0.773$ and $a_2 = 1.27$.

where V_m is the corresponding volume fraction associated with material m and diffusion coefficient κ_m in the cell. We expect the harmonic average to always underestimate the effects of multimaterial heat flow, and it therefore provided a useful lower bound. The numerical FLAG results of Ref. [33] indeed show that the arithmetic mean emphasizes larger values of the diffusion coefficient, while the harmonic mean emphasizes smaller values. For the grid alignment in Fig. 111, the arithmetic average has the effect of extending the material zone and its value of κ beyond the material boundary into the overlapping computational cell proportional to the volume fraction. For the harmonic average, the effect is to decrease the value of κ to zero in the overlapping computational cell, and the work of Ref. [33] illustrates the xRAGE calculations that first bore this out. There are two other diffusion options in FLAG, thin mesh [34] and static condensation [35], which are based on more detailed principles. For example, one starts with the volume fractions of each material in mesh elements, and recreates the material interfaces in the elements using interface reconstruction methods. The mesh is subdivided along the interfaces making sure that the final polyhedral mesh is conforming. Finally, the diffusion equation is solved on the subdivided mesh containing all single material elements. This method is reported to be second order in Ref. [35].

3) The Planar Sandwich Test Problem: We now describe a few details of the planar sandwich of Ref. [33], with the problem definition being illustrated in Fig. 112. We work in arbitrary temperature units, and our starting point will be the 2D heat equation in rectangular coordinates,

$$\frac{\partial T}{\partial t} \quad = \quad \kappa \nabla^2 T = \kappa \left[\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} \right] T \ , \tag{48}$$

where $T = T(x, y, t)$ is the temperature field at position (x, y) and time t, and κ is the (constant) heat diffusion coefficient. We restrict the domain of the problem to be a square with sides of length L, *i.e.* $(x, y) \in [0, L] \times [0, L]$. The test problem is a 2D heat flow problem in x-y rectangular coordinates, consisting of three parallel material layers aligned vertically along the y-direction in a sandwich-like configuration, running from $y = 0$ to $y = L$ along the y-axis. The x-axis extends over $x = 0$ to $x = L$, forming square with sides of length L. The outer two layers do not conduct heat ($\kappa = 0$), while the inner layer is heat conducting with $\kappa > 0$, and runs between $x = a_1$ and $x = a_2$, forming a sandwich of conducting and non-conducting materials.

We do not set κ to be identically zero in numerical simulations; instead instead we use $\kappa = 10^{-12}$ as an approximation to zero in the outer two nonconducting regions. However, the exact solution to the planar sandwich requires that κ be identically zero in the outer layers of the sandwich, giving no heat conduction in those regions, with heat flowing only in the central region. We should therefore think of the condition $\kappa = 10^{-12}$ as a limiting procedure in which $\kappa = \epsilon$ with $\epsilon \to 0^+$. With

```
from exactpack.solvers.heat import PlanarSandwich

solver = PlanarSandwich(TB=1, TT=0,  L=2)

L = 2
y = np.linspace(0.0, L, 1000)
t0 = 1.0
t1 = 0.2
t2 = 0.1
t3 = 0.01
t4 = 0.001

soln0 = solver(y, t0)
soln1 = solver(y, t1)
soln2 = solver(y, t2)
soln3 = solver(y, t3)
soln4 = solver(y, t4)

soln0.plot('temperature')
soln1.plot('temperature')
soln2.plot('temperature')
soln3.plot('temperature')
soln4.plot('temperature')
```

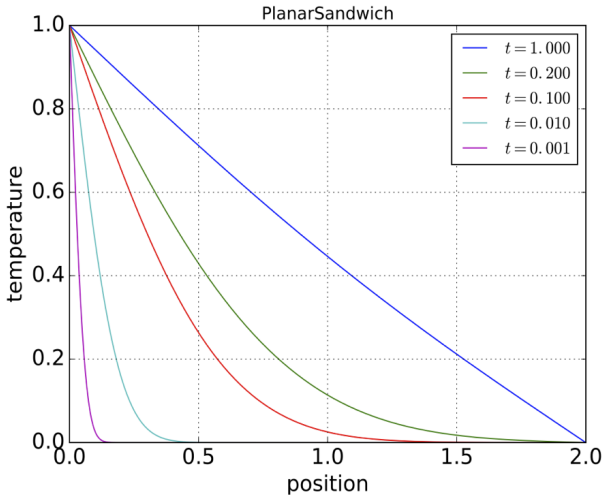


Fig. 113: The Planar Sandwich in ExactPack: PlanarSandwich(T1=1, T2=0, L=2, Nsum=1000). The temperature profile is plotted at times $t = 1, 0.2, 0.1, 0.01$, and 0.001. The BC's are $T(0) = 1$, T(L)=0, and the IC is $T_0 = 0$. The diffusion constant is $\kappa = 1$, the length of the rod is $L = 2$, and we have summed over 1000 terms in the series. The ExactPack scripting commands that produce the plot are to the left.

much more work, one can construct an exact solution for the case in which $\kappa \neq 0$ in the outer two materials, and this solution is currently being added to ExactPack. For now, we shall only consider the case in which $\kappa \to 0^+$ in the outer two regions.

In the original variant of the planar sandwich of Ref. [33], the initial temperature of the system is taken to vanish, so that $T(x, y, t=0) \equiv T_0(x, y) = 0$ for $(x, y) \in (0, L) \times (0, L)$. The temperature boundary condition along the lower $y = 0$ boundary is taken to be $T(x, y=0) = T_1$, with $0 \leq x \leq L$. In principle, T_1 could be a function of x, but the simplest route to an exact solution is let T_1 be a constant. Similarly, we take the temperature boundary condition along the upper $y = L$ boundary to be $T(x, y = L) = T_2$. The left- and right-hand sides of the rectangle are taken to be insulating, *i.e.* the temperature flux in the x-direction at the far left- and right-ends vanishes, $\partial_x T(x = 0, y) = 0$ and $\partial_x T(x = L, y) = 0$. Note that when $T_1 > T_2$, the Second Law of Thermodynamics ensures that heat flows upward from $y = 0$ to $y = L$. Since the upper and lower boundary conditions are uniform in the x-direction, the heat flow is uniform along the y-direction, and the problem reduces to 1D flow. These symmetry arguments reduce the 2D planar sandwich to a 1D heat flow problem oriented along the y-direction, with the heat flow equation in the central region $(x, y) \in (a_1, a_2) \times (0, L)$ reducing to

$$\frac{\partial T}{\partial t} \quad = \quad \kappa \, \frac{\partial^2 T}{\partial y^2} \; . \tag{49}$$

We have represented the 1D temperature profile as a function only of y, so that $T = T(y, t)$, with the boundary conditions $T(0, t) = T_1$ and $T(L, t) = T_2$. We will refer to these BC's as the *lower* and *upper* boundary conditions, respectively. The initial condition becomes $T(y, 0) \equiv T_0(y) = 0$. The exact analytic solution for this problem was presented in Ref. [33], and takes the form

$$T(y, t) \quad = \quad T_1 + \frac{(T_2 - T_1)\, y}{L} + \sum_{n=1}^{\infty} B_n \, \sin(k_n \, y) \, e^{-\kappa \, k_n^2 t} \tag{50}$$

$$k_n \quad = \quad \frac{n\pi}{L} \qquad \text{and} \qquad B_n = \frac{2T_2 \, (-1)^n - 2T_1}{n\pi} \; . \tag{51}$$

Figure 113 is a plot of the y-profile of the planar sandwich solution, along with some of the corresponding ExactPack commands necessary to produce the plot.

4) Convergence Studies of the Planar Sandwich: We now turn to the convergence analysis of the planar sandwich in FLAG. We take the sandwich to be a square region aligned along the x- and y-directions with sides of length $L = 2$. A numerical grid

is formed by dividing the x and y directions into N equal segments, forming squares with sides of length $\Delta x = \Delta y = L/N$, as illustrated in Fig. 111. In our numerical simulations, we take the number of cells to be

$$N = 5\,,\ 10\,,\ 20\,,\ 40\,,\ 80\,,\ 160\,,\ 320\,,640\ , \tag{52}$$

and for $L = 2$, the squares have sides of length

$$\Delta x = \Delta y = 0.4\,, 0.2\,,\ 0.1\,,\ 0.05\,,\ 0025\,,\ 0.0125\,,\ 0.00625\,,\ 0.003125\ . \tag{53}$$

We will occasionally consider a finer grid with $N = 800$, which corresponds to $\Delta x = \Delta y = 0.0025$. For the purposes of this first study, we take the inner conducting region of the sandwich to be determined by $0.25 \leq x \leq 1.25$. In subsequent studies, we find it convenient to restrict the conducting region to $0.2773 \leq x \leq 1.27$, which is both an off-set and a re-scaling of the first region. This renders the effects of harmonics in the frequency sums harmless, and the corresponding plots are continuous.

The purpose of this section is to investigate the numerical solutions, including grid convergence, when the multimaterial cells are handled using (i) an arithmetic average, (ii) a harmonic average, (iii) thin mesh, and (iv) static condensation. Figure 114 illustrates various FLAG runs of the *planar sandwich* test problem under the specified initial conditions.

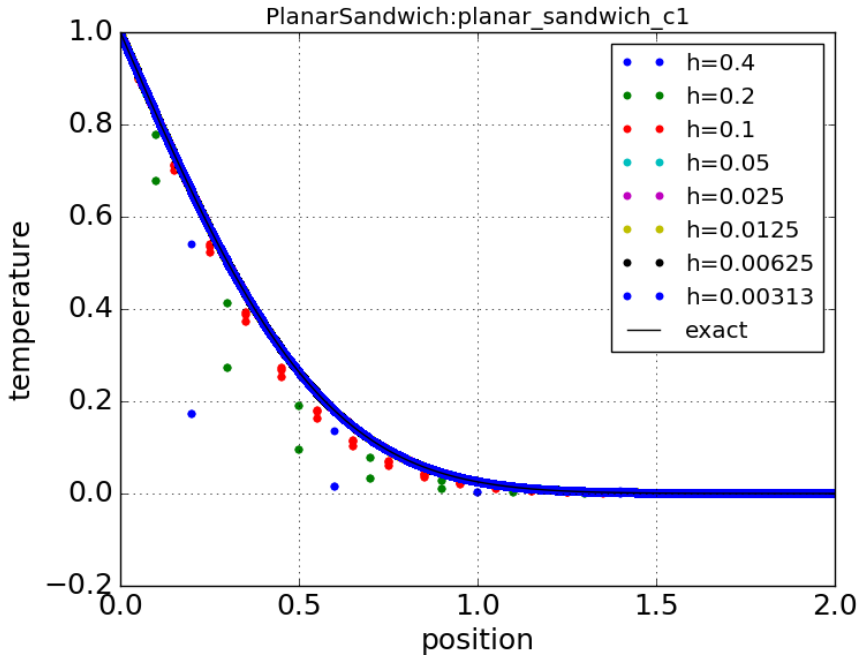


Fig. 114: FLAG runs for the planar sandwich test problem using the arithmetic average for multimaterial cells. The sandwich is take to be a square region with sides of length $L = 2$, aligned along the x- and y-axes, each of which are divided into N equal length cells. For computational purposes, the number of cells is take to be $N = 5, 10, 20, 40, 80, 160, 320, 640$, or in terms of grid resolution, $\Delta x = \Delta y = 0.4, 0.2, 0.1, 0.05, 0025, 0.0125, 0.00625, 0.003125$. We take the heat flux to be $\kappa = 1$ inside the central vertical strip determined by $0.75 \leq x \leq 1.25$, and we take κ to vanish outside this region, *i.e.* $\kappa = 0$ for $0 \leq x < 0.75$ and $1.25 < x \leq 2$. For every point on the numerical grid, (x_i, y_i, T_i), we have plotted (y_i, T_i). In the continuum limit, the solution becomes independent of x_i, and indeed, as the grid resolution increases, the spread in the points decreases, and they lie closer to the analytic solution.

5) The Arithmetic and Harmonic Averages: In this section, we are interested in the arithmetic and harmonic averaging techniques employed in FLAG for multimaterial cells. We perform FLAG runs at the resolutions (52) and (53). We take the heat flux to be constant and uniform, $\kappa = 1$, inside the central vertical strip. In the first set of numerical simulations we take the heat conduction as $0.75 \leq x \leq 1.25$; and we take $\kappa = 10^{-12}$ within $0 \leq x < 0.25$ and $1.25 < x \leq 2$. The horizontal coordinate extends over $0 \leq y \leq 2$. The numerical solution is returned on a square grid, and consists of points (x_i, y_i) with corresponding temperature T_i, where $i = 1, \cdots, N_{\text{numeric}}$. In the figure, we have plotted each point as (y_i, T_i), projecting out the value of x_i. Consequently, for each (y_i, T_i) pair, there are multiple points in the plot corresponding to T_i and x_i, and they become more closely spaced as the resolution increases, until they apear to lie on top of one another at the highest resolutions. Figure 115 illustrates the corresponding convergence analysis using ExactPack, where we have plotted the L_1 norm against

the values of Δy on a log-log scale. The central region lies in $0.75 \leq x \leq 1.25$, and the mesh refinements coincide with this region on the 4^{th} iteration. The discontinuity at $\Delta y = 0.05$ arises because the grid spacing coincides with the material boundary on the 4^{th} grid refinement.

By choosing the conduction strip to be offset by $0.2773 \leq x \leq 1.27$, we can eliminate the discontinuity, since the reduction in grid spacing by factors of two never aligns with a material boundary. The resulting convergence analysis is continuous, as illustrated in Fig. 116, and the convergence rate is order $p = 1$. Except for the offset in the conduction region, the grid parameters are exactly the same as before, with eight uniform grids given by (52) and (53). Since the central region is shifted to $0.773 \leq x \leq 1.27$, the mesh refinements are never commensurate with this region. Figure 117 illustrates the corresponding convergence analysis on the shifted grid using the harmonic average. As one can see, the arithmetic and harmonic averages are quantitatively similar. The shifted conduction region will be our default for the arithmetic and harmonic mean algorithms in all following analyses for the planar sandwich.

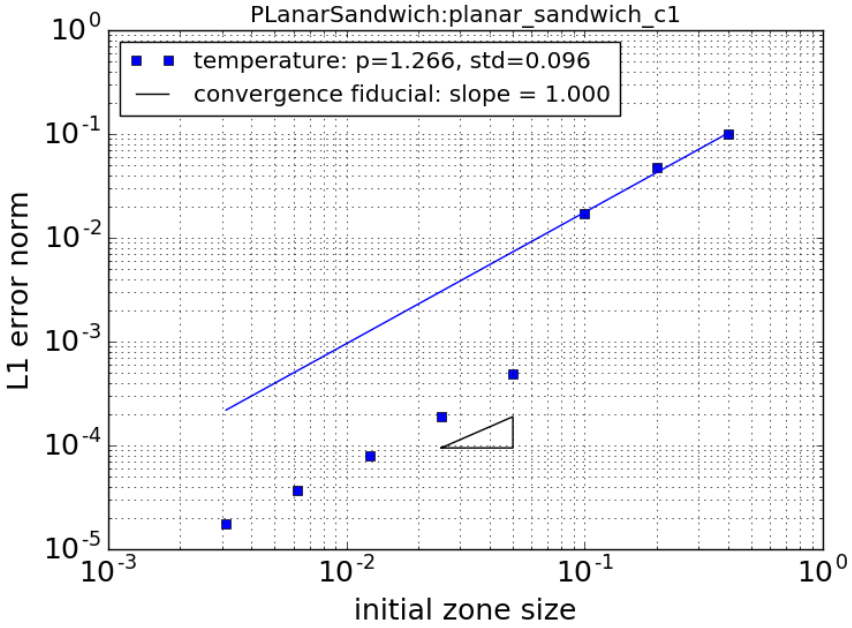


Fig. 115: Convergence analysis for the planar sandwich in FLAG using the arithmetic average algorithm to handle multimaterial cells. The *length* of the physical region is $L = 2$, and we have used 8 uniform grids with $N = 5, 10, 20, 40, 80, 160, 320, 640$, or $\Delta y = 0.4, 0.2, 0.1, 0.05, 0025, 0.0125, 0.00625, 0.003125$. The central region lies in $0.75 \leq x \leq 1.25$, and the mesh refinements coincide with this region on the 4^{th} iteration.

6) Thin Mesh and Static Condensation: We now turn to the mesh reconstruction algorithms of FLAG, namely *thin mesh* [34] and *static condensation* [35]. We take the heat conducting region to be the determined $0.75 \leq x \leq 1.25$. Figure 118 show the FLAG solutions for thin mesh plotted against the analytic solution, while Figure 119 shows the convergence analysis for thin mesh. Note that, as opposed the the arithmetic and harmonic averages, there is no discontinuity on the grid delimited by $0.75 \leq x \leq 1.25$. Note that Fig. 119 shows that thin mesh starts out at 2^{nd} order for coarser meshes, and then becomes 1^{st} order as the mesh in refined. Fig. 121 shows the corresponding study for static condensation, which exhibits quantitatively similar scaling.

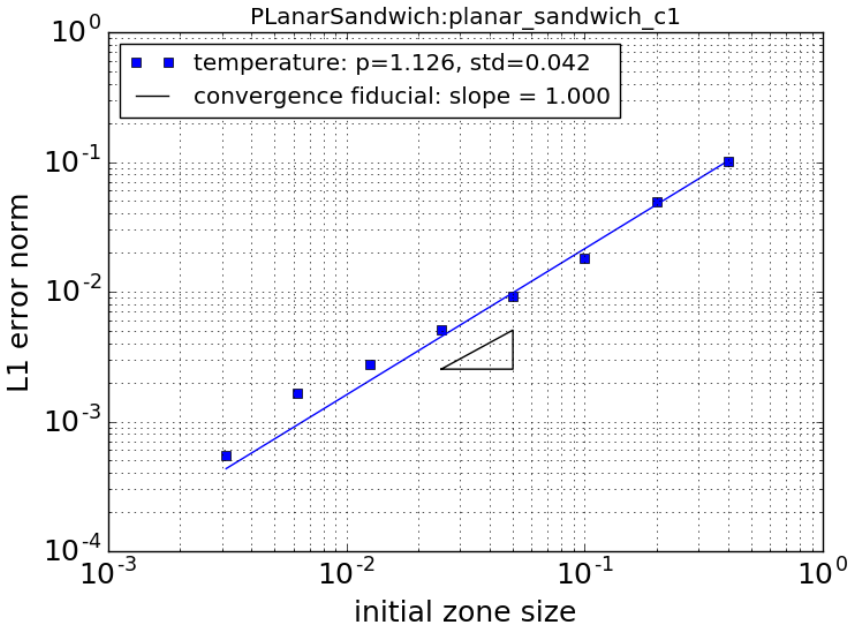


Fig. 116: Convergence analysis for the planar sandwich in FLAG using the arithmetic average treatment of multimaterial cells. The length of the region is $L = 2$, and we have used 8 uniform grids with $N = 5, 10, 20, 40, 80, 160, 320, 640$, or $\Delta y = 0.4, 0.2, 0.1, 0.05, 0025, 0.0125, 0.00625, 0.003125$. The central region is shifted to $0.773 \leq x \leq 1.27$, and the mesh refinements are never commensurate with this region.

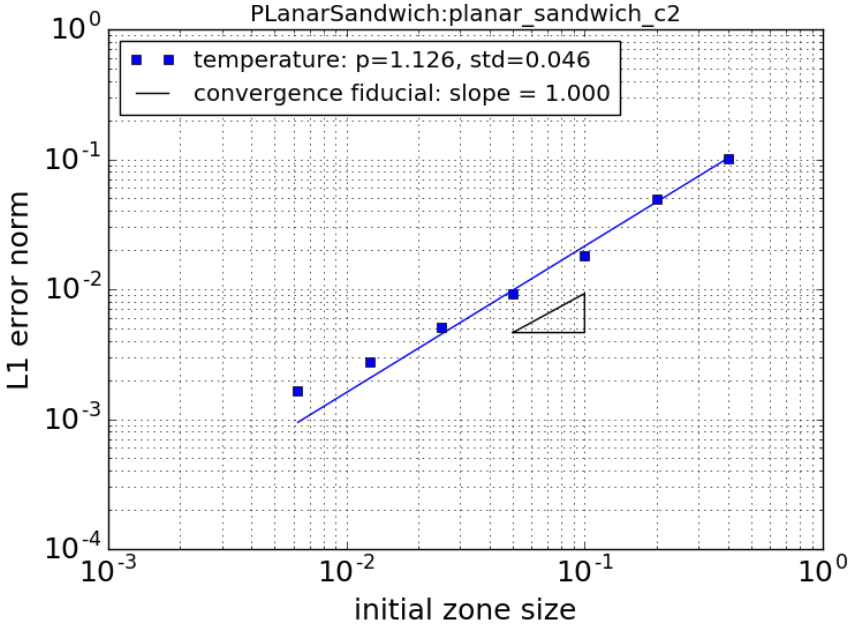


Fig. 117: As in Fig. 116, except that the harmonic average is used for multimaterial cells.

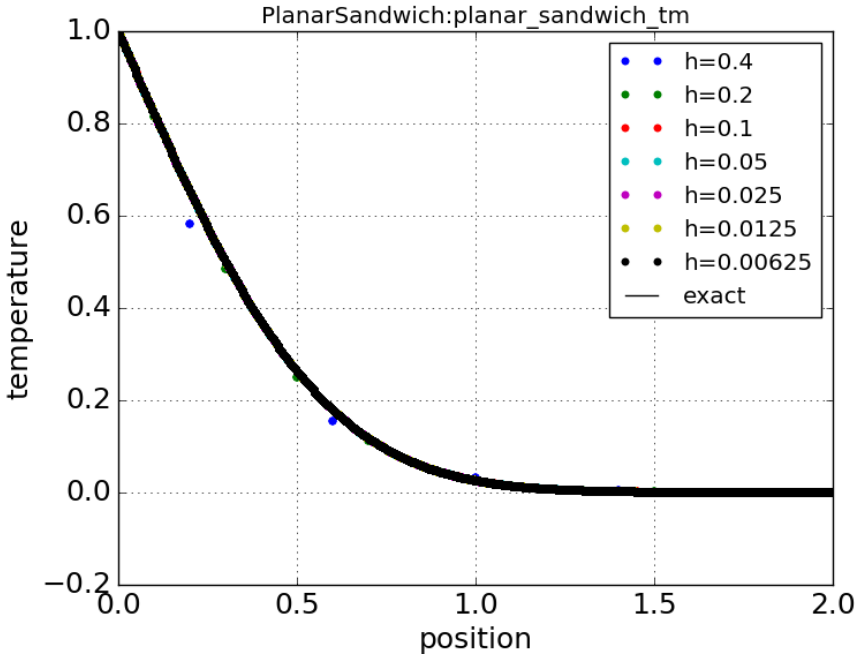


Fig. 118: The thin mesh option for multimaterial zones. FLAG runs for the *planar sandwich* test problem on a square region with sides of length $L = 2$. We take the heat flux to be $\kappa = 1$ inside the central region determined by $0.75 \leq x \leq 1.25$, and we take to it vanish outside this region, *i.e.* $\kappa = 0$ for $0 \leq x < 0.75$ and $1.25 < x \leq 2$. The number of computational regions is $N = 5, 10, 20, 40, 80, 160, 320, 640$, or in terms of grid resolution, $\Delta x = \Delta y = 0.4, 0.2, 0.1, 0.05, 0025, 0.0125, 0.00625, 0.003125$. The value of the diffusion constant κ in the multimaterial cells along the material boundary has assigned using using the thin mesh in the multimaterial homogenization algorithm.

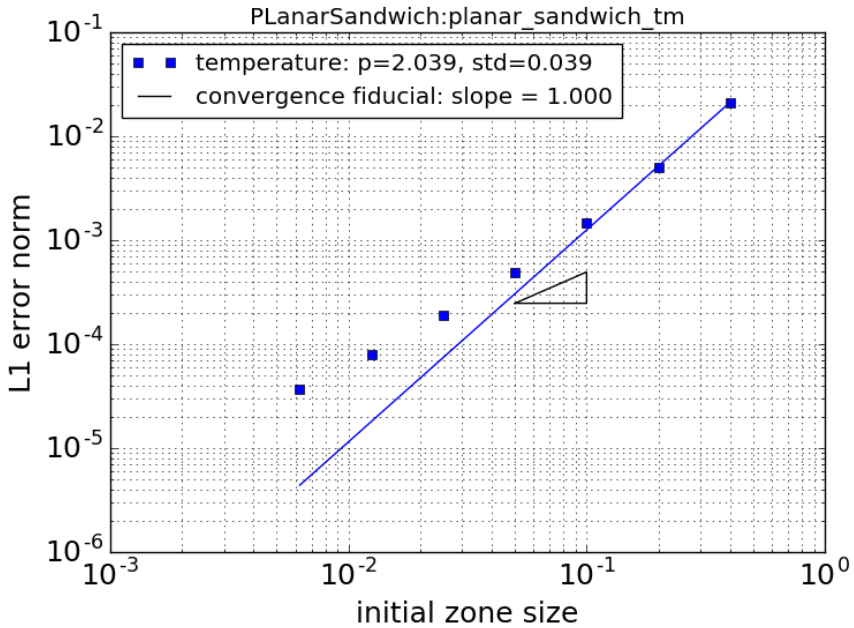


Fig. 119: Convergence analysis for the planar sandwich in FLAG using the multimaterial *thin mesh* algorithm on 8 uniform grids: $L = 2$ with $N = 5, 10, 20, 40, 80, 160, 320, 640$, or $dx = 0.4, 0.2, 0.1, 0.05, 0025, 0.0125, 0.00625, 0.003125$.

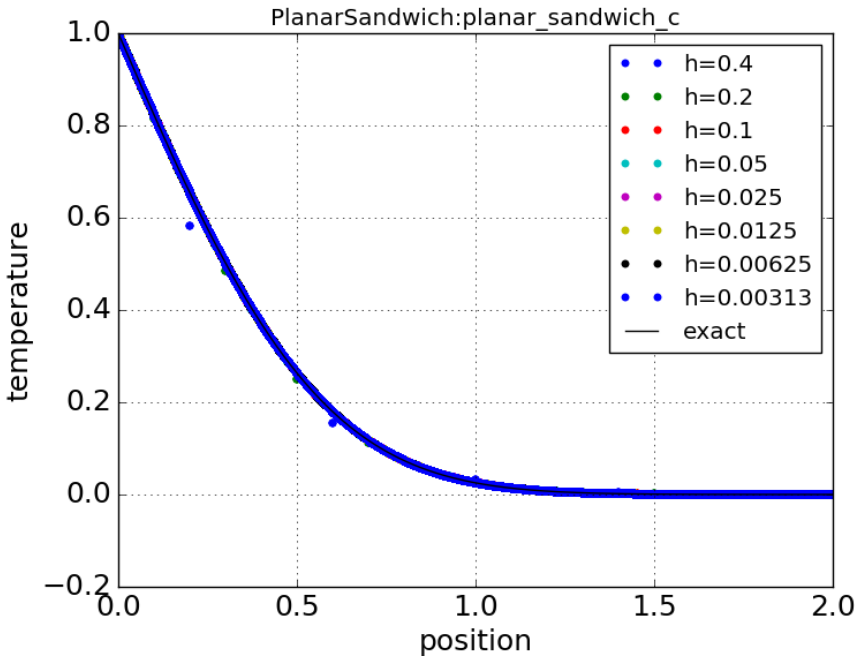


Fig. 120: As in Fig. 118, except for static condensation

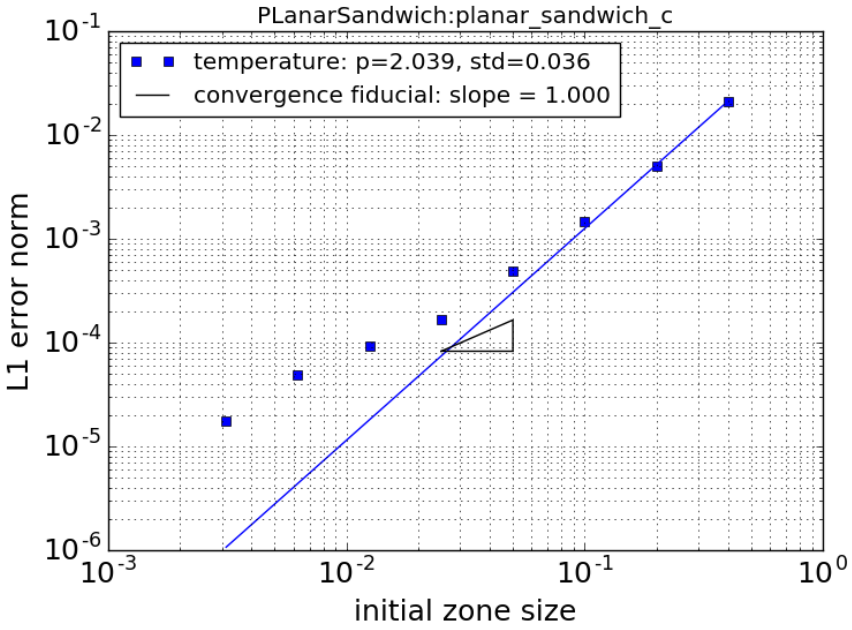


Fig. 121: Convergence analysis for the planar sandwich in FLAG using the multimaterial mesh of static condensation on 8 uniform grids: $L = 2$ with$N = 5, 10, 20, 40, 80, 160, 320, 640$, or $\Delta y = 0.4, 0.2, 0.1, 0.05, 0025, 0.0125, 0.00625, 0.003125$.

ACKNOWLEDGMENT

The authors would like to thank the many LANL colleagues who contributed to this effort, including especially Miles Buechler, Josh Childress, Ralph Menikoff, James Kamm, and Ann Kaul. This work was sponsored by the Advanced Simulation and Computing program within the National Nuclear Security Administration of the U.S. Department of Energy under contract DE-AC52-06NA25396.

REFERENCES

[1] R.L. Singleton, D.M. Israel, S.W. Doebling, C.N. Woods, A. Kaul, J.W. Walter, M.L. Rogers, "Exactpack documentation," Los Alamos National Laboratory, Los Alamos, New Mexico 87545, Los Alamos Report LA-UR-16-23260 (rev. 1), 2017. [Online]. Available: http://github.com/lanl/exactpack

[2] D. M. Israel, "Lightweight simulation process and data management documentation," Los Alamos National Laboratory, Software Manual LA-UR-17-28306, 2017.

[3] ——, "Software infrastructure for V&V," Los Alamos National Laboratory, Report LA-UR-17-28311, 2017.

[4] S. W. Doebling, "Standardized definitions for code verification test problems," Los Alamos National Laboratory, Report LA-UR-17-28269, 2017, this supersedes LA-UR-14-20418.

[5] J. M. Ferguson, J. E. Morel, and R. B. Lowrie, "The equilibrium-diffusion limit for radiation hydrodynamics," *Journal of Quantitative Spectroscopy & Radiative Transfer*, vol. 202, pp. 176–186, 2017.

[6] ——, "Nonrelativistic grey S_n-transport radiative-shock solutions," *High Energy Density Physics*, vol. 23, pp. 95–114, 2017.

[7] O. Delchini, M., J. C. Ragusa, and J. M. Ferguson, "Viscous regularization of the full set of nonequilibrium-diffusion grey radiation-hydrodynamic equations," *International Journal for Numerical Methods in Fluids*, vol. 85, pp. 30–47, 2017.

[8] S. W. Doebling, "The escape of high explosive products: An exact-solution problem for verification of hydrodynamics codes," *Journal of Verification, Validation and Uncertainty Quantification*, vol. 1, no. 4, 2016.

[9] C. N. Woods, "Verification assessments using the verification test suite," Los Alamos National Laboratory, Report LA-UR Pending, 2017.

[10] Performance Test Code Committee 60: Verification and Validation in Computational Solid Mechanics, *ASME V&V 10-2006: Guide for Verification and Validation in Computational Solid Mechanics*. New York, New York, 10016: The American Society for Mechanical Engineers, 2006.

[11] V&V20 Committee - Verification and Validation in Computational Fluid Dynamics and Heat Transfer, *ASME V&V 20-2009: Standard for Verification and Validation in Computational Fluid Dynamics and Heat Transfer*. New York, New York, 10016: The American Society for Mechanical Engineers, 2009.

[12] W. L. Oberkampf and C. J. Roy, *Verification and Validation in Scientific Computing*. Cambridge CB2 8RU, United Kingdom: Cambridge University Press, 2010.

[13] J. W. Banks, T. Aslam, and W. J. Rider, "On sub-linear convergence for linearly degenerate waves in capturing schemes," *Journal of Computational Physics*, vol. 227, no. 14, pp. 6985–7002, 2008. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0021999108002088

[14] R. Menikoff and J. J. Quirk, "Versatile, User-Friendly EOS Package," Tech. Rep. LA-UR-01-6222, 2001.

[15] R. Menikoff, "Verification test of the SURF and SURFplus models in xRage," Los Alamos National Laboratory, Full Paper Report LA-UR-16-23636, 2016.

[16] ——, "Verification Test of the SURF and SURFplus Models in xRage: Part II," Los Alamos National Laboratory, Full Paper Report LA-UR-16-24352, 2016.

[17] ——, "Verification test of the SURF and SURFplus models in xRage: Part III affect of mesh alignment," Los Alamos National Laboratory, Full Paper Report LA-UR-16-26317, 2016.

[18] J. R. Kamm, J. S. Brock, S. T. Brandon, D. L. Cotrell, B. M. Johnson, P. Knupp, W. J. Rider, T. G. Trucano, and V. G. Weirs, "Enhanced verification test suite for physics simulation codes," Los Alamos National Laboratory Lawrence Livermore National Laboratory Sandia National Laboratories, Report LA-14379, 2008. [Online]. Available: http://permalink.lanl.gov/object/tr?what=info:lanl-repo/lareport/LA-14379

[19] E. J. Caramana, M. J. Shashkov, and P. P. Whalen, "Formulations of artificial viscosity for multi-dimensional shock wave computations," *Journal of Computational Physics*, vol. 144, pp. 70–97, 1998.

[20] F. Brezzi, A. Buffa, and K. Lipnikov, "Mimetic finite differences for elliptic problems," *ESAIM: Mathematical Modelling and Numerical Analysis*, vol. 43, p. 277295, 2009.

[21] J. Campbell and M. Shashkov, "A tensor viscosity using a mimetic finite difference algorithm," *Journal of Computational Physics*, vol. 172, pp. 739–765, 2001.

[22] N. Morgan, K. Lipnikov, D. Burton, and M. Kenamond, "A Lagrangian staggered grid Godunov-like approach for hydrodynamics," *Journal of Computational Physics*, vol. 259, pp. 568–597, 2014.

[23] T. O. Masser, J. G. Wohlbier, and R. B. Lowrie, "Shock wave structure for a fully ionized plasma," *Shock Waves*, vol. 21, pp. 367–381, 2011.

[24] R. B. Lowrie and R. M. Rauenzahn, "Radiative shock solutions in the equilibrium diffusion limit," *Shock Waves*, vol. 16, pp. 445–453, 2007.

[25] R. B. Lowrie and J. D. Edwards, "Radiative shock solutions with grey nonequilibrium diffusion," *Shock Waves*, vol. 18, pp. 129–143, 2008.

[26] D. Mihalas and B. W. Mihalas, *Foundations of Radiation Hydrodynamics*, ser. Dover Books on Physics Series. Mineola, N.Y.: Dover, 1999.

[27] A. S. Eddington, *The Internal Constitution of the Stars*, 1926.

[28] F. G. Blake, Jr., "Spherical wave propagation in solid media," *Journal of the Acoustical Society of America*, vol. 24, no. 2, pp. 211–215, 1952, cited By :45 Export Date: 20 June 2016 Correspondence Address: Blake, F.G.; California Research Corporation, La Robra, California, United States. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-84953656355&partnerID=40&md5=e2eeb9fb80c3d777f5abaceed154aec6

[29] S. Hunter, "The propagation of spherically symmetric disturbances in ideally plastic materials," in *Proceedings of the Conference on Properties of Materials at High Rates of Strain, London 1957*, 1957, pp. 147–155.

[30] J. R. Kamm, "Analysis Of The Blake And Hunter Problems With The Sage Code," Los Alamos National Laboratory, Report LA-UR-08-06050 ; LA-UR-08-6050, 2008, sponsor: DOE Classification Level (full-text): U Repository ID: info:lanl-repo/lareport/LA-UR-08-06050 Dataset: LA-SAFE1. [Online]. Available: http://permalink.lanl.gov/object/tr?what=info:lanl-repo/lareport/LA-UR-08-06050

[31] J. S. Brock, "Elastic & perfectly plastic solid dynamic test problems," Los Alamos National Laboratory, Report LA-UR-11-03420, 2011.

[32] M. Shashkov, "Algorithms for improved solution robustness, accuracy and mesh adaptivity," Los Alamos, Tech. Rep. LA-UR-15-23913, 2015.

[33] A. Dawes, C. Malone, and M. Shashkov, "Some new verification test problems for multimaterial diffusion on meshes that are non-aligned with material boundaries."

[34] R. Garimella and K. Lipnikov, "Solution of the diffusion equation in multi-material domains by subdivision of elements along reconstructed interface," 2011.

[35] K. Evgeny, K. Yuri, and M. Shashkov, "New algorithm for solving multi-material diffusion problems on mesh non-aligned with material interfaces," 2016.