LA-UR-09-09-01964

Approved for public release; distribution is unlimited.

Title: NUMERICAL UNCERTAINTY IN COMPUTATIONAL ENGINEERING AND PHYSICS

(Manuscript)

Author(s): François M. Hemez, Los Alamos National Laboratory, X-3

Intended for: 2nd International Conference on Uncertainty in Structural Dynamics, Sheffield, United Kingdom, June 15-17, 2009

This page is left blank intentionally.

2nd International Conference on Uncertainty in Structural Dynamics (USD)
Sheffield, United Kingdom, June 15-17, 2009, Paper number USD-09-041

NUMERICAL UNCERTAINTY IN COMPUTATIONAL ENGINEERING AND PHYSICS

François M. Hemez

X-Division (X-3), Los Alamos National Laboratory
Los Alamos, New Mexico 87545, U.S.A.

E-mail: hemez@lanl.gov

ABSTRACT. Obtaining a solution that approximates ordinary or partial differential equations on a computational mesh or grid does not necessarily mean that the solution is accurate or even *"correct."* Unfortunately assessing the quality of discrete solutions by questioning the role played by spatial and temporal discretizations generally comes as a distant third to test-analysis comparison and model calibration. This publication is contributed to raise awareness of the fact that discrete solutions introduce ***numerical uncertainty***. This uncertainty may, in some cases, overwhelm in complexity and magnitude other sources of uncertainty that include experimental variability, parametric uncertainty and modeling assumptions. The concepts of consistency, convergence and truncation error are overviewed to explain the articulation between the exact solution of continuous equations, the solution of modified equations and discrete solutions computed by a code. The current state-of-the-practice of code and solution verification activities is discussed. An example in the discipline of hydro-dynamics illustrates the significant effect that meshing can have on the quality of code predictions. A simple method is proposed to derive bounds of solution uncertainty in cases where the exact solution of the continuous equations, or its modified equations, is unknown. It is argued that numerical uncertainty originating from mesh discretization should always be quantified and accounted for in the overall uncertainty *"budget"* that supports decision-making for applications in computational physics and engineering.

KEYWORDS: Code and solution verification, mesh refinement, numerical uncertainty.

1. INTRODUCTION

Modeling and Simulation (M&S) has established itself in the last 30 years as the approach of choice for solving engineering and physics problems. Examples of applications where M&S has already had a profound impact on the safety and reliability of engineered products include airfoil design, the performance of automobiles during crashes and structural health monitoring. Questions such as *"how to absorb shocks and avoid passenger injury during an automobile collision?"* or *"how to mitigate the effects of earthquakes in urban environments?"* are answered using M&S on a routine basis. More recently, M&S has become the tool of choice for analyzing trends in global climate dynamics and assess the efficacy of effect mitigation strategies.

It has become evident in recent years that the most difficult challenge to obtain credible answers when applying M&S to a specific problem is to manage prediction uncertainty. Broadly speaking, the main three sources of uncertainty are physical experimentation (measurements or ***observations***), modeling (***assumptions***) and solution procedure (***numerics***). How to estimate measurement uncertainty is well understood by experimentalists. Replication can be combined to the propagation of uncertainty through the experimentation and signal processing steps to arrive at bounds of measurement uncertainty. Likewise tools have been developed to address modeling uncertainty. They include parametric methods such as the forward propagation or inverse inference of probability density functions, model calibration and non-parametric methods

such as stochastic modeling (polynomial chaos, random matrices, etc.). The proceedings of the 1st International Conference on Uncertainty in Structural Dynamics offer an excellent overview of the state-of-the-practice in the discipline of structural dynamics [1].

In this publication we focus our attention on the third kind, numerical uncertainty, because it seems to be the least well understood of the three. Numerical (or solution) uncertainty results from the fact that the ordinary or partial differential equations being modeled in a computer code are discretized and solved with a computational method. The resulting solutions always satisfy a set of equations that are different from the original, continuous equations. In short, solution error refers to the difference between these continuous and discrete solutions. If the exact solution of the continuous equations is unknown, then, the ***error*** becomes an ***uncertainty*** that should be modeled and accounted for in the overall uncertainty *"budget"* of the numerical simulation. How to do so rigorously is, however, somewhat unclear.

This manuscript starts with a brief review of techniques developed to verify the accuracy and performance of *"black box"* computer codes using mesh refinement. Because the focus is on non-intrusive methods, error estimation is not addressed [2]. We start in section 2 with a discussion of modified equation analysis. In section 3, the three concepts of consistency, convergence and truncation error are overviewed to explain the articulation between the exact solution of continuous equations, solution of modified equations and discrete solutions obtained from a code. Section 4 discusses the current practice of performing mesh refinement to assess solution accuracy. The significant effect that meshing can have on the quality of predictions is illustrated in section 5 with a hydro-dynamics test problem analyzed with a Lagrangian code.

The last part of this work discusses the quantification of solution uncertainty when the exact solution of the continuous equations is unknown. This applies to all practical situations where a computer code is used because equations cannot be solved *"by hand."* A simple derivation is proposed in section 6 to estimate bounds of solution uncertainty when scalar-valued predictions are analyzed. The quantification of solution uncertainty is illustrated with the non-linear Burgers equation and the extension to multi-dimensional fields, $y \in \Re^N$, is mentioned and illustrated.

We conclude that numerical uncertainty may, in some cases, be the *"900-pound gorilla"* that dwarfs other types of uncertainty. This is particularly acute in applications that involve unstable dynamics, sub-grid models or strong interactions between dissimilar time and length scales. The good news is that a theory is available to formalize solution uncertainty and approaches are being developed, such as the one proposed in this work, to rigorously quantity it.

2. THE MODIFIED EQUATION OF A NUMERICAL METHOD

To understand where solution uncertainty comes from, and motivate the importance of studying and quantifying it, we give a brief overview of the concept of ***modified equation***. In short, the equation represents the equation verified by the approximated solution obtained with a numerical method on the basis of a computational mesh or grid.

The starting point is the realization that, when a system of partial differential equations (or ordinary differential equations) is solved numerically by a computer code, the solution obtained is different from the solution of the original equations. In fact the solution obtained does not even solve the same equations! To illustrate the discussion, we assume that we are solving partial differential equations in 1D geometry that can be written as a set of conservation laws such as:

$$\frac{\partial y^{Exact}(x;t)}{\partial t} + \frac{\partial F(y^{Exact}(x;t))}{\partial x} = S(x;t), \qquad (1)$$

where $y^{Exact}(x;t)$ is the exact solution, $F(\bullet)$ denotes the flux term and $S(x;t)$ is a source or forcing function that drives the dynamics of the system. The solution y^{Exact} of equation (1) can be scalar-valued or represent a multi-dimensional field, $y^{Exact} \in \Re^N$. The fact that equation (1) is restricted to 1D geometry, and somewhat simplistic, does not restrict our discussion. This equation represents, for example, a simplified version of the Navier-Stokes equations in fluid dynamics, Euler equations that govern gas dynamics, finite element formulation in solid mechanics, or evolution of species in a chemical reaction.

The exact solution y^{Exact} is continuous in the sense that it is solution to a set of equations that are defined at every single point (x;t) in space-time. y^{Exact} may be physically discontinuous, such as in the case of a shocked flow around an airfoil, but the basic idea is that it defines a mathematical function that is known everywhere in the domain. Often, the exact solution cannot be computed analytically because closed-form solutions can only be obtained in the case of simple equations, linear models and simple initial and boundary conditions. One then resorts to implementing a numerical method to discretize equation (1) on a computational mesh or grid.

Implementation of a numerical method yields a numerical solution that can be written as:

$$y_k^n = y(k\cdot\Delta x; n\cdot\Delta t)\,, \tag{2}$$

where Δx and Δt are increments in space and time assumed, for clarity, to be constant. An approximation such as defined in equation (2) is characteristic of finite differencing schemes for which values of the solution field are sought at specific, discrete points in space-time. Note that this notation does not preclude examination of the case where a finite volume method (or any other method) is implemented because, in these cases, one can define the discrete solution as:

$$y_k^n = \int_{[t_n;t_{n+1}]}\left(\int_{\Omega_k} y(x;t)\cdot\varphi_k(x)\cdot dx\right)\cdot\varphi_n(t)\cdot dt\,, \tag{3}$$

where $\Phi_k(x)$ and $\Phi_n(t)$ are weighting functions or *"shape functions"* in space and time defined, respectively, over the k^{th} volume Ω_k of the domain and n^{th} interval of time integration $[t_n; t_{n+1}]$. A finite volume method would, for example, use piece-wise linear interpolation $\Phi_k(x) = \alpha_k + \beta_k\cdot x$ and constant interpolation in time $\Phi_n(t) = 1$. Another example of the formalism described in equation (3) would be the finite element method where values of the solution field are obtained, for example, at integration points within each element of the computational mesh.

To obtain the numerical solution y_k^n, one needs to solve a system of discretized equations that, following our 1D example, can be written as an equation that looks something like:

$$\frac{y_k^{n+1}-y_k^n}{\Delta t}+\frac{F_{k+1/2}^n-F_{k-1/2}^n}{\Delta x}=S_k^n\,. \tag{4}$$

Equation (4) implements a discretization similar to those illustrated in equations (2-3) for the source term in the right-hand side and F_k^n denotes discretization of the flux, $F_k^n = F(y_k^n)$. The equation illustrates a discretized equation where, for example, the Euler forward differencing is implemented in time (fully explicit) and a trapezoidal differencing rule is implemented in space.

One observes that the discretized equation *"appears"* similar to the original, continuous equation (1) but the similarity is misleading. In fact the two are different. It can be shown that approximations y_k^n converge to the (exact) solution of a modified equation [3] that has the form:

$$\frac{\partial\hat{y}}{\partial t}+\frac{\partial F(\hat{y})}{\partial x}=S+\left(\frac{\partial^2\hat{y}}{\partial t^2}\cdot\Delta t+\frac{\partial^3\hat{y}}{\partial x^3}\cdot\Delta x^2+\frac{\partial\hat{y}}{\partial t}\cdot\frac{\partial^2\hat{y}}{\partial x^2}\cdot\Delta t\cdot\Delta x^2+\text{H.O.T.}\right), \tag{5}$$

where the dependency of variables $\hat{y}$ and S on space and time is omitted for simplicity. This example is notional and the correct form of the modified equation depends on the combination

of properties of the continuous equation and numerical method used to solve it. Two important factors in deciding what the modified equation looks like are, first, the mathematical properties of equations (elliptic, hyperbolic or parabolic?) and, second, the functional form of the flux (linear or non-linear?). The essential point is that the solution $\hat{y}$ of equation (5) is ***different*** from the solution y^{Exact} of equation (1). In fact, solutions $\hat{y}$ and y^{Exact} do not even solve the same equation, at least, not until $\Delta x = 0$ and $\Delta t = 0$ exactly … which never happens in practice!

Understanding that the accuracy and performance of a numerical method are governed by its modified equation, and not just the properties of a numerical algorithm implemented in the code, is essential to study solution error and quantify numerical uncertainty. Truncation error is overviewed next because the performance of a numerical method is generally assessed by studying its asymptotic properties. The discussion leads to the concepts of convergence of a discrete solution and consistency of a numerical method.

3. THE LINK BETWEEN CONVERGENCE, CONSISTENCY AND TRUNCATION

Truncation error refers to the difference between the solution of the continuous equations and the discrete solution obtained with a computational mesh. Since we have seen that we are dealing with two different continuous equations, this begs the question of which *"exact"* solution should be used to define truncation error. ***Should one use solution y^{Exact} of equation (1) or $\hat{y}$ of equation (5)?*** Common practice is to refer back to the exact solution y^{Exact} even though one could argue that solution $\hat{y}$ is a more appropriate choice since the modified equation governs what is *"truly"* being solved in the code (at least, as long as $\Delta x \neq 0$ and $\Delta t \neq 0$).

For completeness, it is mentioned that recent results on *"finite scale equations"* pursue this logic to discover that a modified equation includes, in addition to truncation error, a contribution from physically-motivated terms [4-5]. Discriminating truncation error from physics is, of course, key to define more meaningful convergence testing protocols. It also offers a path forward to construct better numerical methods.

Common practice, however, is not to consider the modified equation, nor the *"finite scale equation"* of Reference [5], but to define truncation error as the difference between the exact solution y^{Exact} of equation (1) and the discrete solution y_k^n of equation (4). This difference can be expressed in the sense of a global norm that is appropriate for the problem:

$$\varepsilon(\Delta x;\Delta t) = \left\| y^{Exact}(x;t) - y_k^n(\Delta x;\Delta t) \right\|. \tag{6}$$

An appropriate choice of norm $\|\bullet\|$ in the context, for example, of the finite element method is the weak norm of the equations of motion. Applied to the *"toy"* example of equations (1-5) and assuming, first, linearity of the flux term (that is, $F(y) = \alpha \cdot y$ where α is a constant coefficient or operator) and, second, that the limit of infinite space-time resolution is taken (that is, $\Delta x \rightarrow 0$ and $\Delta t \rightarrow 0$), one can relate truncation error to the modified equation as follows:

$$\varepsilon(\Delta x;\Delta t) = \frac{\partial^2 y^{Exact}}{\partial t^2}\cdot\Delta t + \frac{\partial^3 y^{Exact}}{\partial x^3}\cdot\Delta x^2 + \frac{\partial y^{Exact}}{\partial t}\cdot\frac{\partial^2 y^{Exact}}{\partial x^2}\cdot\Delta t\cdot\Delta x^2 + H.O.T. \tag{7}$$

The significance of these steps is to illustrate that the error of a properly implemented numerical method can be driven to zero by increasing the spatial resolution and reducing the time step. This is generally achieved by running the calculation on a finer computational mesh because the controls of explicit or implicit time stepping algorithms usually subordinate the time step Δt to the size Δx_{Min} of the smallest zone, cell or element in the mesh.

The above derivations show why it is important to assess the behavior of truncation error: it is our *"gateway"* into understanding if the numerical method behaves according to expectation.

Equation (7) suggests, for example, that our numerical method should be first-order accurate in time and second-order accurate in space because the leading terms of the expansion are proportional to powers Δt and Δx^2, respectively. Not meeting these expectations may indicate an implementation deficiency or coding mistake.

Strictly speaking, equation (7) is correct ***only*** at the limit $\Delta x \to 0$ and $\Delta t \to 0$. This regime is where truncation error $\varepsilon(\Delta x;\Delta t)$ is dominated by the lower-order terms of the (potentially infinite) expansion. This motivates studying the asymptotic behavior using a simple equation such as:

$$\varepsilon(\Delta x;\Delta t) \approx \beta \cdot \Delta x^{p}, \tag{8}$$

where the pre-factor coefficient β is assumed to be constant and the exponent p denotes the rate-of-convergence of the numerical method, as $\Delta x \to 0$. Running, for example, a finite element calculation using second-order elements should provide an observed rate of p = 2 under mesh refinement. The series expansion (8) is often restricted to spatial convergence because, in most calculations, temporal resolution is *"slaved"* to spatial convergence. References [6-7] discuss the analysis of combined, space-time convergence.

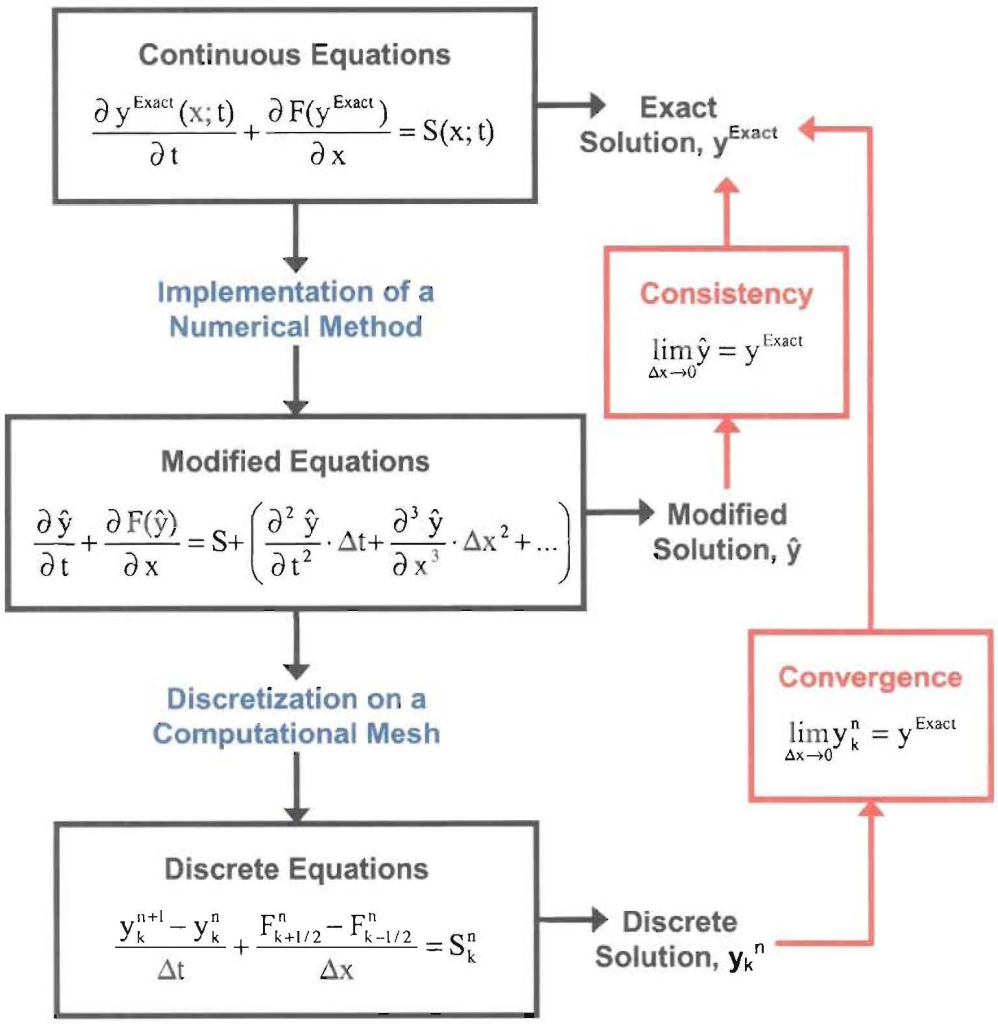

Figure 1. Articulation between exact, modified and discrete equations.

To find the governing equation of truncation error $\varepsilon(\Delta x;\Delta t)$, such as shown in equation (7), we have implicitly assumed ***consistency*** of the numerical method. This property states that the modified equation is consistent with the original equation. This can be captured by writing that:

$$\lim_{\Delta t\to 0,\Delta x\to 0} \hat{y} = y^{Exact}. \tag{9}$$

(Strictly speaking, *"consistency"* is a property of a numerical method while *"convergence,"* see below, is a property of a discrete solution.) The property captured in the model of truncation error (8), on the other hand, expresses ***convergence***. Convergence states that the discrete solution approaches the exact solution under mesh refinement:

$$\lim_{\Delta t\to 0,\Delta x\to 0} y_k^n = y^{Exact}. \tag{10}$$

The connection between exact, modified and discrete equations and the flow of information required to assess the properties of consistency and convergence are illustrated graphically in Figure 1. For completeness, it should also be mentioned that the concepts of consistency and convergence are linked through the famous equivalence theorem of Peter Lax who established in 1954 that ***"stability + consistency ↔ convergence"*** for linear systems where $F(y) = \alpha \cdot y$ in equation (1), see Reference [8]. It means that stability and consistency of the numerical method are necessary and sufficient to establish convergence of the discrete solutions.

Developing an understanding of the link between consistency, convergence and truncation error is necessary to discuss how to improve the practice of code and solution verification. In the next section, mesh refinement is overviewed because it is the mechanism to generate datasets that enable the study of asymptotic convergence (*"are we converged yet?"* question of Figure 1) and the quantification of solution uncertainty, as discussed in section 6.

4. WHAT IS A MESH REFINEMENT STUDY USEFUL FOR?

We briefly overview the state-of-the-practice of mesh refinement because it is the technique commonly accepted to investigate the asymptotic behavior of truncation error. Mesh refinement refers to running the same problem on two, possibly more, meshes to accumulate enough instances of equation (8) and solve for its two unknowns (β; p). It can then be verified that the observed rate-of-convergence matches the formal order of accuracy of the numerical method.

Assuming that the exact solution y^{Exact} of the continuous equations is known analytically, which implies that mesh refinement is applied to simple test problems, two calculations can be performed using coarse and fine discretizations. The discrete solution computed on the coarse-size mesh is labeled $y_k^n(\Delta x_C)$ where Δx_C denotes a characteristic size of the coarse mesh. Likewise, the discrete solution obtained from the refined mesh is labeled $y_k^n(\Delta x_F)$ where Δx_F is a characteristic size of the fine mesh. Two values of truncation error (6) can then be computed as:

$$\varepsilon(\Delta x_C) = \left\| y^{Exact}(x;t) - y_k^n(\Delta x_C) \right\| \quad \text{and} \quad \varepsilon(\Delta x_F) = \left\| y^{Exact}(x;t) - y_k^n(\Delta x_F) \right\|, \tag{11}$$

where dependency on time discretization Δt is omitted for simplicity. It can be easily verified that the solution to equation (8) is obtained as:

$$p = \log\left(\frac{\varepsilon(\Delta x_C)}{\varepsilon(\Delta x_F)}\right) / \log(R), \tag{12}$$

where R denotes the refinement ratio, that is, $R = \Delta x_C/\Delta x_F > 1$. A closed-form solution can also be found for the regression pre-factor β of equation (8). The observed rate-of-convergence is compared to the theoretical order of accuracy of the method to verify the appropriateness of its

algorithmic implementation (*"is there any code deficiency that prevents reaching* $p = p^{Theory}$*?"*) and assess solution accuracy (*"should* Δx *be decreased to improve solution accuracy?"*).

More details about the application of mesh refinement to code verification can be obtained from References [6-7, 9-13]. Practical details are discussed in Reference [12] and, in particular, what the best strategy may be to compute the error differences of equation (11) when the exact, coarse-mesh and fine-mesh solutions are expressed on different computational grids.

We now shift our attention to the case where the exact solution of the continuous equations is unknown. This is referred to as calculation verification (or solution verification). The procedure outlined in equations (11-12) cannot be applied directly because the solution error $\varepsilon(\Delta x)$ cannot be calculated if y^{Exact} is unknown. The practice most often encountered in computational physics and engineering is to restrict the analysis to the special case of monotonic convergence of scalar-valued solutions. The exact solution y^{Exact} is replaced by an (unknown) reference solution $y^{Reference}$ that becomes a third unknown of the problem. In addition equation (8) is specialized to scalar-valued responses to make it possible to go through the derivations *"by hand."*

This simplified variant of equations (6) and (8) becomes:

$$y^{Reference} = y(\Delta x) + \beta \cdot \Delta x^{p} + O(\Delta x^{p}). \tag{13}$$

Because equation (13) features three unknowns ($y^{Reference}$; β; p), a minimum of three equations are needed. These equations are provided by the discrete solutions obtained from coarse-mesh (Δx_C), medium-mesh (Δx_M) and fine-mesh (Δx_F) calculations. It means that the same problem must be analyzed a minimum of three times, using three different meshes. It can be verified that the rate-of-convergence is solution to the following non-linear equation:

$$p \cdot \log(R_{MF}) + \log(1 - R_{CM}^{p}) - \log(1 - R_{MF}^{p}) = \log\left(\frac{y(\Delta x_M) - y(\Delta x_C)}{y(\Delta x_F) - y(\Delta x_M)}\right), \tag{14}$$

where R_{CM} denotes the refinement ratio from coarse-to-medium meshes ($R_{CM} = \Delta x_C/\Delta x_M$) and R_{MF} is the refinement ratio from medium-to-fine meshes ($R_{MF} = \Delta x_M/\Delta x_F$). There is no closed-form solution to equation (14), except in the case of a constant refinement where $R_{CM} = R_{MF} = R$. This special case yields the well-known solution:

$$p = \log\left(\frac{y(\Delta x_M) - y(\Delta x_C)}{y(\Delta x_F) - y(\Delta x_M)}\right) / \log(R). \tag{15}$$

Finally, solutions $y(\Delta x_C)$, $y(\Delta x_M)$ and $y(\Delta x_F)$ can be extrapolated to a common reference $y^{Reference}$ that is the best estimate, given the three discrete solutions available, of the solution that would be predicted if the calculation could be run at *"infinite resolution,"* $\Delta x = 0$. The extrapolation is:

$$y^{Reference} = y(\Delta x_F) + \frac{y(\Delta x_F) - y(\Delta x_M)}{R^{p} - 1}, \tag{16}$$

where p denotes the observed rate-of-convergence from equation (14) or (15). Knowing p and $y^{Reference}$, the regression coefficient β can be back-calculated from equation (13).

An example of performing mesh refinement to verify that an algorithm yields the theoretical order of accuracy of the numerical method is presented in the next section. Going back to the question *"what is mesh refinement useful for?,"* we see that it is the mechanism that generates datasets to verify the asymptotic behavior of truncation error. We will show in section 6 that mesh refinement also enables the estimation of bounds of solution uncertainty when the exact solution of the continuous equation, or the solution of the modified equation, is unknown.

5. IS THE MESH THE BIGGEST *"KNOB"* OF YOUR SIMULATION?

The computational mesh or grid is generally not thought of as being a *"knob,"* or calibration variable, of a numerical simulation that can be adjusted to produce a desired result. But the fact that discrete solutions are only approximations of the exact solution and that, as seen in section 2, they do not even verify the same equations begs the question of understanding the influence of a mesh on the quality of the solution obtained. So we ask: ***do you know the extent to which your mesh influences the accuracy and overall quality of your numerical solution?***

To illustrate that this is indeed a sensible question, and that situations may arise where the computational mesh may very well be the most influential knob, we discuss the simulation of the Noh problem. It is a code verification test problem that assesses the ability of a hydro-dynamics method to convert kinetic energy into internal energy. Figure 2 (left) illustrates that the Noh problem is a uniformly convergent flow towards the origin of the domain. The initial condition creates an instantaneous discontinuity (or shock) that expands outwards. The conservation laws nevertheless possess an exact solution $y^{Exact}(x;t)$ [14]. The exact solution is plotted in Figure 2 (right) for the density field using a 1D, spherical coordinate system.

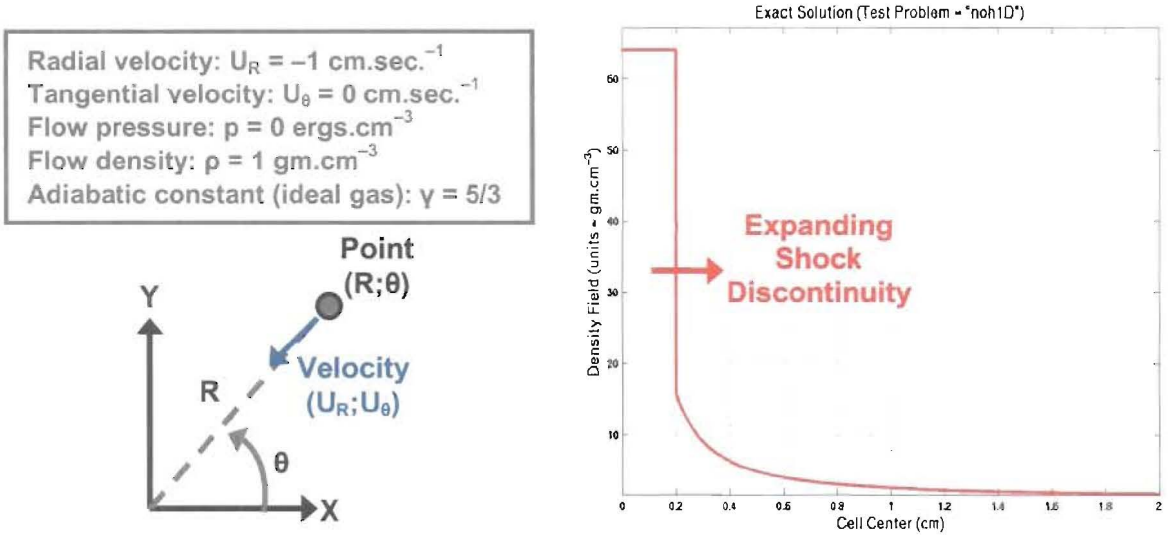

Figure 2. Initial condition of the Noh test problem (left) and exact solution y^{Exact} (right).

The code used for this example is developed at Los Alamos for research and development in hydro-dynamics methodologies and algorithms. It solves the Euler equations of gas dynamics using a second-order accurate finite volume method defined in a Lagrangian frame-of-reference and capable of handling arbitrary geometries. The Noh problem is discretized in 2D geometry and a half-symmetry domain is analyzed. Figure 3 depicts two discretizations analyzed. The one on the left of Figure 3 is analyzed first. It features a flow-aligned grid while the second one (on the right) does not due to the presence of a square mesh inserted at the center of the domain.

The Noh problem is first solved using the flow-aligned mesh of Figure 3 (left) and in pure Lagrangian mode. This means that zones are allowed to move with the flow without any mesh adaptation. Figure 4 plots the values of solution error $||y^{Exact}(x;t) - y_k^n(\Delta x)||_1$ for the density field as a function of mesh size Δx. Truncation error is defined with the L^1 norm, a choice that is consistent with the analysis of a discontinuous solution. Values of $||y^{Exact}(x;t) - y_k^n(\Delta x)||_1$ and Δx are shown on logarithmic scales such that equation (8) should become a straight line of slope $p^{Theory} = 1$. (The second-order accuracy of the finite volume method should degrade to first-order due to the presence of the shock discontinuity.)

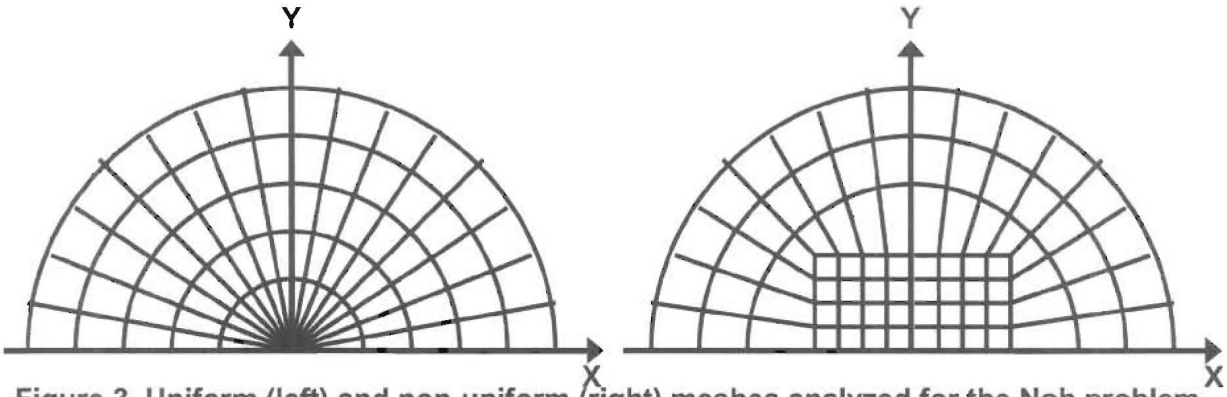

Figure 3. Uniform (left) and non-uniform (right) meshes analyzed for the Noh problem.

Figure 4 shows that, as expected, the first-order asymptotic behavior is recovered when the same calculation of the Noh problem is performed with four different mesh sizes. The observed rate-of-convergence is equal to p = 0.987, which confirms that the Lagrangian hydro-dynamics method implemented in the code performs satisfactorily on a flow-aligned mesh.

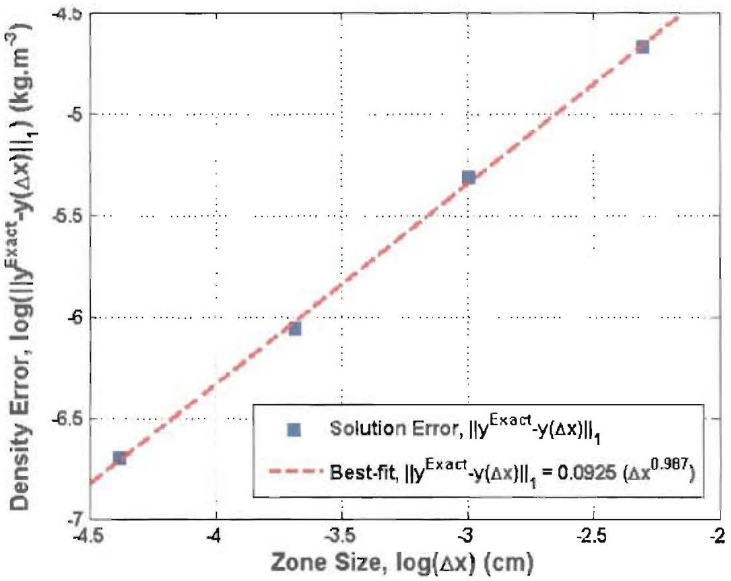

Figure 4. Asymptotic convergence of the finite volume method for the Noh problem.

The analysis is repeated next with the non-uniform mesh illustrated in Figure 3 (right). Mesh adaptation is turned on during the calculation. Adaptation is needed to help the hydro-algorithm negotiate the *"corners"* of the imprinted box as the material flows inwards. Running the problem with either mesh should not make any difference because the mesh simply provides a support to express the solution. It should not, ***in itself***, be a significant factor that influences accuracy.

The Arbitrary Lagrangian Eulerian (ALE) algorithm adapts the mesh in an attempt to better *"follow"* the flow and avoid highly distorted zones that would pollute the numerical quality of the solution. ALE is turned on at every cycle of time integration. After a full hydro-step is completed that solves the conservation laws, ALE performs a rezoning step where a new mesh is obtained that minimizes zone distortion. This first step is followed by a second one where solution fields expressed on the old (distorted) mesh are re-mapped onto the new (somewhat more uniform) mesh. This completes the cycle and the calculation advances to the next time step, $t_{k+1} = t_k + \Delta t$.

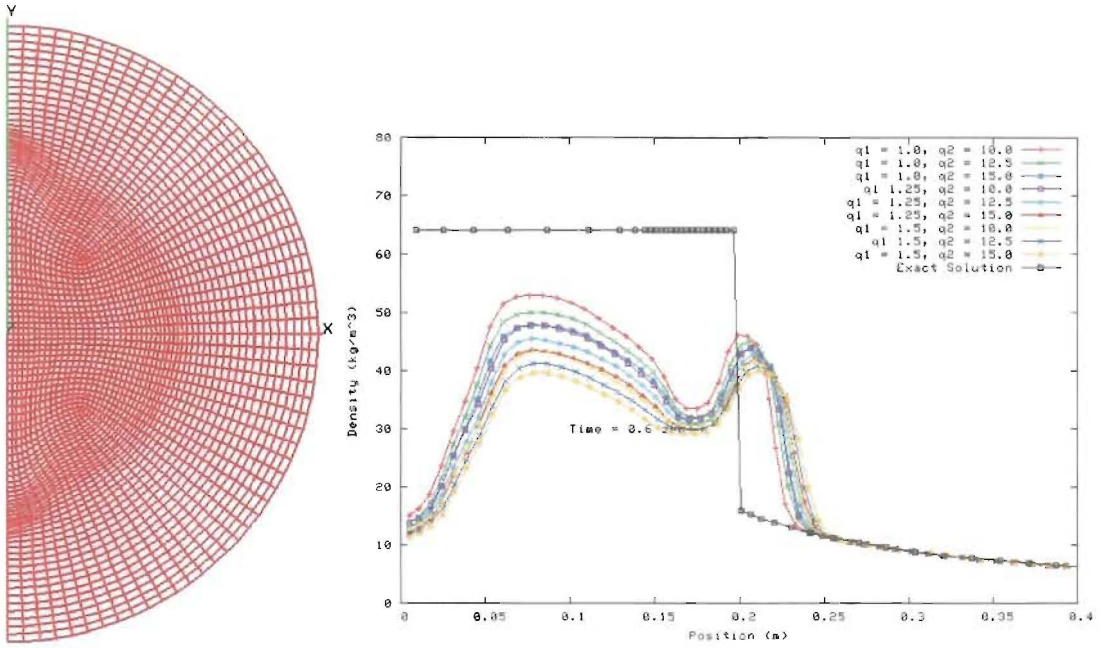

Figure 5. ALE results obtained with the type-A rezoning and type-I viscosity model.

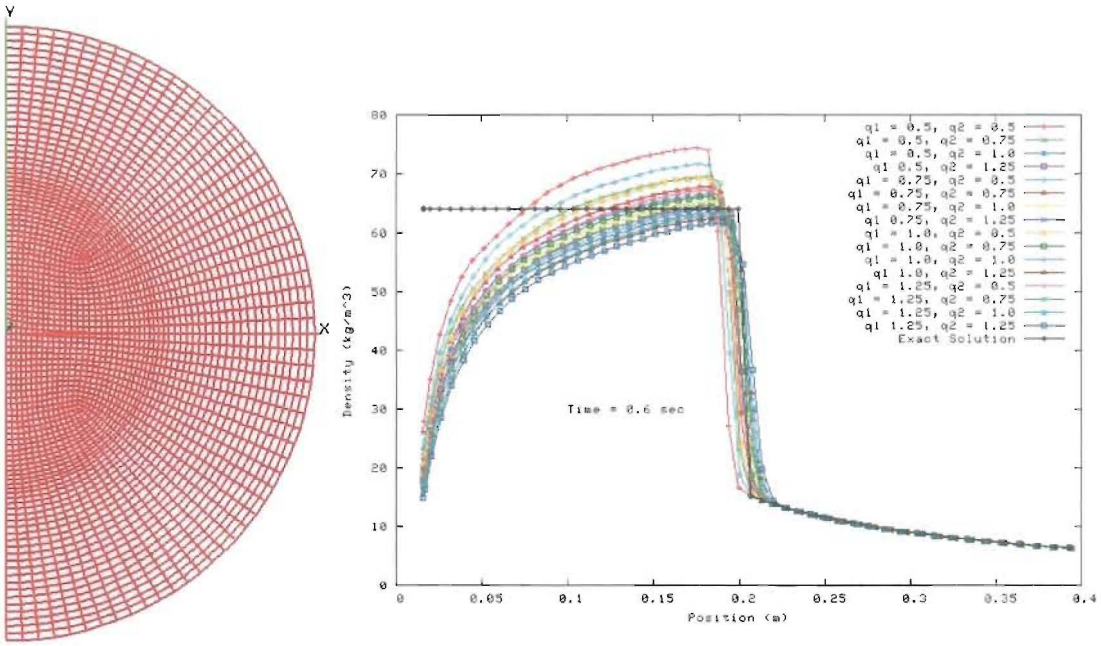

Figure 6. ALE results obtained with the type-A rezoning and type-II viscosity model.

Figures 5-7 show, on the left, snapshots of the deformed mesh at time t = 0.6 sec. and, on the right, comparisons between discrete and exact solutions. The exact solution is plotted with a black, solid line. The colored lines represent multiple discrete solutions obtained by varying the amount of artificial viscosity allowed in the calculation. Figures 5-7 compare discrete solutions obtained with different combinations of two rezoning algorithms (labeled type-A and type-B) and

two models of artificial viscosity (labeled type-I and type-II). Details about these calculations are available from Reference [15].

Figures 5 and 6 clearly demonstrate that poor-quality meshes are produced by the type-A rezoning algorithm which, in turn, leads to poor accuracy at, and behind, the shock front. These meshes deteriorate the quality of discrete solutions to a point where it becomes difficult to trust the predictions. On the other hand, the solution shown in Figure 7 and obtained by running with the combination of type-B rezoning algorithm and type-II model of artificial viscosity matches the expected accuracy. Simply by varying the type of computational mesh as shown in Figure 3, we observe vastly different performances. Of course, the rezoning and re-mapping steps of the ALE strategy can also influence the numerical quality of solutions. But the role of mesh adaptation is primarily in avoiding mesh tangling and guaranteeing the *"survivability"* of the calculation.

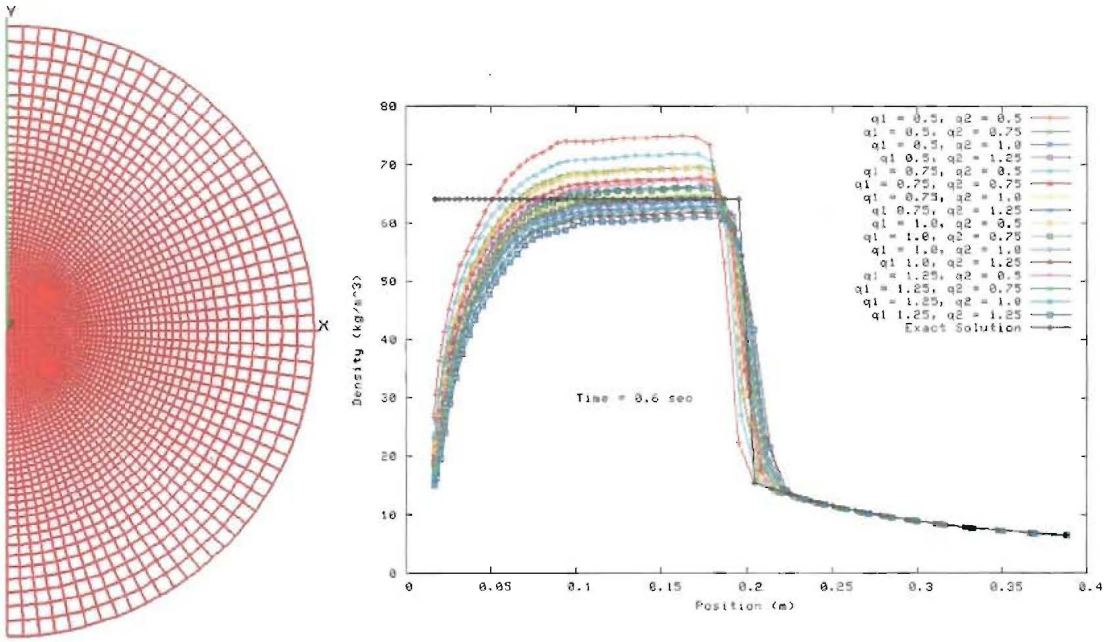

Figure 7. ALE results obtained with the type-B rezoning and type-II viscosity model.

This example illustrates that meshing can play a critical role in delivering good-quality or poor-quality solutions. This may be a somewhat extreme example but the same remains true of calculations performed with other numerical methods, such as finite element analysis. The point we are making is that we may not always realize the extent to which our computational meshes or grids may be "knobs" in our simulations, just like parameters of viscosity models or material models can be calibrated to match physical measurements. If the effect of meshing is unknown, then, the uncertainty originating from running with a given mesh should be rigorously quantified to increase the confidence placed in predictions. This topic is further addressed next.

6. A PROPOSAL FOR ESTIMATING BOUNDS OF SOLUTION UNCERTAINTY

In sections 2 and 3, we have briefly reviewed the theoretical framework available to assess the accuracy of solutions calculated by a numerical method implemented in a code. This has lead to an overview of code and calculation verification activities in section 4. We now turn our attention to open areas of research and development and, more specifically, the quantification of solution uncertainty.

Techniques available to study asymptotic convergence are based on the practice of mesh refinement to accumulate discrete solutions by running the calculation with different mesh or grid sizes. These methods assume that the exact solution y^{Exact} of the continuous equations is known, which helps to define truncation error and carry out derivations. But what happens when the exact solution is unknown?

We have seen in equations (13-16) that solution verification can be achieved using an extrapolated solution $y^{Reference}$ to replace the *"true-but-unknown"* solution y^{Exact}. Solution error, such as $|y^{Exact}(x;t) - y_k^n(\Delta x)|$ can then be replaced by an approximation $|y^{Reference}(x;t) - y_k^n(\Delta x)|$ and the equations can still be solved as long as asymptotic convergence is monotonic and restricted to scalar-valued response quantities. It is our contention, however, that when the *"gold standard"* of an exact solution y^{Exact} is unknown, solution ***error*** becomes solution ***uncertainty***.

When the exact solution is unknown, the best that one can do is bound the solution error of prediction $y_k^n(\Delta x)$ obtained at mesh size Δx. Our goal is, therefore, to arrive at an upper bound of uncertainty defined as:

$$\left| y^{Exact}(x;t) - y_k^n(\Delta x) \right| \le U_k^n(\Delta x), \tag{17}$$

where the exact solution y^{Exact} of the continuous equations is unknown. To render the derivations possible in closed-form, analysis is restricted below to scalar-valued responses. The case of multi-dimensional fields is addressed at the end of the section for $y^{Exact} \in \Re^N$ and $y_k^n \in \Re^N$.

Derivations start by writing error models for solutions obtained with coarse-mesh (Δx_C) and fine-mesh (Δx_F) discretizations. It implies that the calculation must be run twice, on two different meshes. Assuming that these runs are located within the regime of asymptotic convergence, the following equations hold:

$$\begin{cases} y^{Exact} \approx y_k^n(\Delta x_F) + \beta \cdot \Delta x_F^p \\ y^{Exact} \approx y_k^n(\Delta x_C) + \beta \cdot \Delta x_C^p \end{cases}. \tag{18}$$

The well-known triangular inequality $|a| + |b| \ge |a + b|$ is modified as $|x - y| \ge |x| - |y| \ge 0$ (simply use $x = a + b$ and $y = b$) and applied to the difference between two discrete solutions:

$$\underbrace{y_k^n(\Delta x_F) - y_k^n(\Delta x_C)}_{x-y} = \underbrace{\left(y_k^n(\Delta x_F) - y^{Exact}\right)}_{x} - \underbrace{\left(y_k^n(\Delta x_C) - y^{Exact}\right)}_{y}. \tag{19}$$

Using the two instances of equation (18), one can write:

$$\left| y_k^n(\Delta x_F) - y_k^n(\Delta x_C) \right| \ge \left| y^{Exact} - y_k^n(\Delta x_C) \right| - \left| y^{Exact} - y_k^n(\Delta x_F) \right| = \beta \cdot \Delta x_C^p - \beta \cdot \Delta x_F^p, \tag{20}$$

where it is assumed, without loss of generality, that the pre-factor coefficient is positive ($\beta > 0$) and the fine-mesh error is smaller than the coarse-mesh error. Introducing the refinement ratio defined as $R = \Delta x_C / \Delta x_F > 1$, one can re-write the previous equation as:

$$\left| y_k^n(\Delta x_F) - y_k^n(\Delta x_C) \right| \ge \beta \cdot \Delta x_F^p \cdot \left(R^p - 1\right), \tag{21}$$

and recalling from equation (18) that $|y^{Exact} - y_k^n(\Delta x_F)| = \beta \cdot \Delta x_F^p$ for the fine-mesh solution, the term $(\beta \cdot \Delta x_F^p)$ can be eliminated:

$$\frac{\left| y_k^n(\Delta x_F) - y_k^n(\Delta x_C) \right|}{\left(R^p - 1\right)} \ge \left| y^{Exact} - y_k^n(\Delta x_F) \right|. \tag{22}$$

What is arrived at is an upper bound of solution uncertainty for the fine-mesh calculation. It is therefore proposed to define solution uncertainty as:

$$U_k^n(\Delta x_F) = \frac{F_S}{R^p - 1} \cdot \left| y_k^n(\Delta x_F) - y_k^n(\Delta x_C) \right|, \tag{23}$$

where F_S is a user-defined factor of safety ($F_S \geq 1$) used to provide additional conservatism. The analogy between definition (23) and the Grid Convergence Index (GCI) of References [16-18] is noticed. The GCI of a fine-grid calculation is computed as:

$$GCI = \frac{F_S}{R^p - 1} \cdot \left| \frac{y_k^n(\Delta x_F) - y_k^n(\Delta x_C)}{y_k^n(\Delta x_F)} \right| \tag{24}$$

with $1 \leq F_S \leq 3$. Even though the two equations are similar, it is emphasized that the motivation of the GCI is completely different. According to Patrick Roache, the idea behind the GCI is:

> *"... to approximately relate the error obtained by whatever grid convergence study is performed (whatever p and R) to the error that would be expected from a grid convergence study of the same problem, with the same fine grid, using p = 2 and R = 2, i.e. a grid doubling with a 2nd-order method."* ([17], chapter 5, pp. 115.)

Another difference is that the GCI requires a factor of safety $F_S = 3$ to be true to the comparison of mesh refinement studies mentioned above while there is no such constraint for the bound of solution uncertainty defined in equation (23).

In the remainder of this section, the solution uncertainty bound of equation (23) is applied to a simple differential equation to illustrate its performance. The estimation of solution uncertainty bounds for multi-dimensional fields is also briefly illustrated. The equation chosen for analysis is Burgers equation in 1D, Cartesian geometry. Though easy to work with, it is also a non-linear, hyperbolic equation capable of developing discontinuous solutions [19]. The continuous Burgers equation with diffusion added in the right-hand side is defined as:

$$\frac{\partial y}{\partial t}(x;t) + \frac{1}{2} \cdot \frac{\partial y^2}{\partial x}(x;t) = \mu \cdot \frac{\partial^2 y}{\partial x^2}(x;t), \tag{25}$$

where $\mu > 0$. Burgers equation is solved in the domain $-½ \leq (x/L) \leq ½$ using an initial condition defined by an arc-tangent function, $y(x; t=0) = y_0(x) = (L/\pi) \cdot \tan^{-1}(\omega \cdot x)$, and over the time period of $0 \leq t \leq 1$ sec. Values L = 3 m and $\omega = 5$ cm^{-1} are used for numerical application.

Equation (25) is solved in conservation form using the Lax-Wendroff integration scheme implemented as a finite volume method with piece-wise, constant interpolation within each zone [20-21]. This algorithm is second-order accurate when applied to the simulation of continuous solutions. The non-linear, hyperbolic nature of Burgers equation evolves the initial condition $y_0(x)$ in a sharp discontinuity, which deteriorates accuracy of the algorithm to first-order.

Table 1. Results of a grid refinement study for the 1D Burgers equation.

Grid Resolution	Cell Size, Δx	Density Jump, $[y]_S$	Rate-of-convergence, p
Extra-coarse, $y_k^n(\Delta x_{XC})$	5.00 cm	0.9995 gm.cm^{-3}	p = 1.01 (The theory predicts $p^{Theory} = 1$.)
Coarse, $y_k^n(\Delta x_C)$	1.25 cm	1.3083 gm.cm^{-3}	
Medium, $y_k^n(\Delta x_M)$	0.3125 cm	1.3832 gm.cm^{-3}	
Fine, $y_k^n(\Delta x_F)$	0.078125 cm	1.4018 gm.cm^{-3}	
Extrapolation, $y^{Reference}$	$\Delta x \rightarrow 0$	1.4080 gm.cm^{-3}	

Four runs are performed with 60, 240, 960 and 3,840 zones and the results are reported in Tables 1 and 2. The mesh refinement ratio is constant and equal to R = 4. The prediction of interest is the discontinuity jump $[y]_S$ defined as the value of the solution to the left of the

discontinuity minus the value to the right. Table 1 lists the predictions of discontinuity jump $[y]_S$ and the best-fitted rate-of-convergence. A value of p = 1.01 is observed, which is in excellent agreement with the expectation of first-order behavior. The prediction $y^{Reference}$ at *"infinite"* mesh resolution, that is, $\Delta x \rightarrow 0$, is also shown. This extrapolation represents the best approximation, given the discrete solutions available, of the unknown solution of the continuous equations.

Table 2 lists the values of bounds $U_k^n(\Delta x)$ computed from equation (23) and corresponding percentages of solution uncertainty. It can be observed that these values are relatively low, hence, suggesting that the calculation is sufficiently resolved with $\Delta x \leq 0.3125$ cm. In a formal Verification and Validation (V&V) assessment, solution uncertainty would be compared to other types of uncertainty, such as experimental variability or parametric uncertainty, to decide on the best course of action needed to improve prediction accuracy or reduce the *"total uncertainty."*

Table 2. Bounds of solution uncertainty for the 1D Burgers equation (with F_S = 1).

Grid Resolution	Cell Size, Δx	Uncertainty, $U_k^n(\Delta x)$	Percent $U_k^n(\Delta x)/y_k^n(\Delta x)$
Extra-coarse, $y_k^n(\Delta x_{XC})$	5.00 cm	N/A	N/A
Coarse, $y_k^n(\Delta x_C)$	1.25 cm	0.1011 gm.cm^{-3}	7.72%
Medium, $y_k^n(\Delta x_M)$	0.3125 cm	0.0245 gm.cm^{-3}	1.77%
Fine, $y_k^n(\Delta x_F)$	0.078125 cm	0.0061 gm.cm^{-3}	0.43%

Figure 8 summarizes graphically the analysis of Tables 1 and 2. It depicts the predictions of jump discontinuity $[y]_S$ obtained with different mesh resolutions, extrapolated solution $y^{Reference}$ and bounds of solution uncertainty $U_k^n(\Delta x)$. For the purpose of this illustration, the uncertainty bounds plotted are twice the values listed in Table 2, that is, $2 \cdot U_k^n(\Delta x)$.

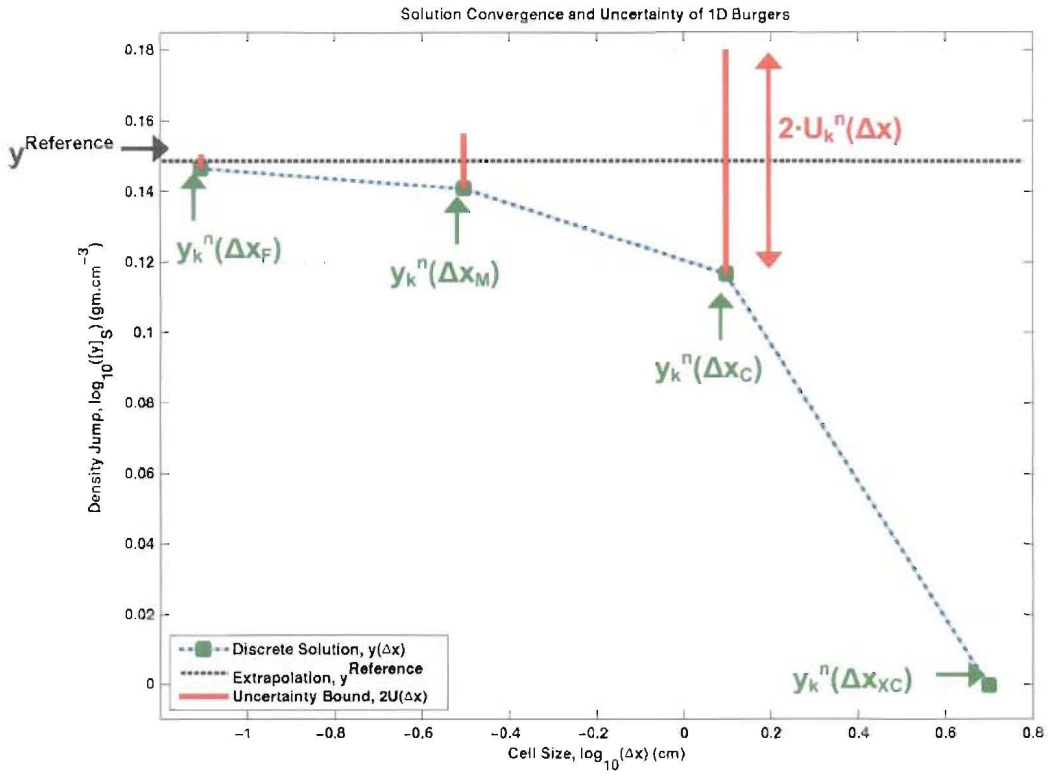

Figure 8. Predictions $[y]_S$ and uncertainty bounds $U_k^n(\Delta x)$ for Burgers equation.

It is clear from Figure 8 that the asymptotic convergence of discrete solutions is monotonic. Even though values $2 \cdot U_k^n(\Delta x)$ are plotted on Figure 8, it can be verified from the tables that the bounds of uncertainty encompass the extrapolated solution: $|y^{Reference} - y_k^n(\Delta x)| \leq U_k^n(\Delta x)$. This makes sense since $y^{Reference}$ provides the best estimate of the exact-but-unknown solution. The practical significance of uncertainty bound $U_k^n(\Delta x)$ is to define an interval where the exact-but-unknown solution y^{Exact} may be located relative to prediction $y(\Delta x)$ at mesh resolution Δx.

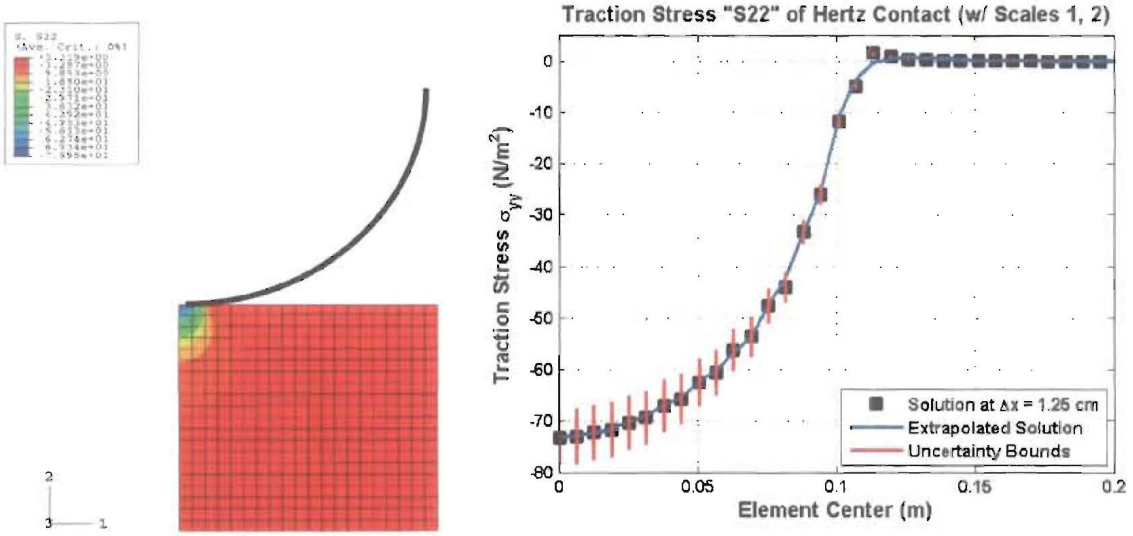

Figure 9. Solution uncertainty for a finite element prediction of stress curve [23].

To conclude we briefly mention that the estimation of solution uncertainty has, so far, been restricted to scalar-valued responses. This makes derivations possible in closed-form. However, predictions from computer codes generally come in the form of multi-dimensional fields such as curves or images. The technique can be generalized for application to multi-dimensional fields, as shown in References [22-24]. The basic idea is to decompose discrete solutions $y_k^n(\Delta x) \in \Re^N$ obtained from a mesh refinement study on a truncated basis of empirical *"modes,"* then, apply equations (17-23) to the ***coordinates*** of this expansion. Solution uncertainty for the entire field, that is, $U_k^n(\Delta x) \in \Re^N$, can then be recomposed using the truncated basis and interval arithmetic to combine the intervals of solution uncertainty for coordinates kept in the expansion.

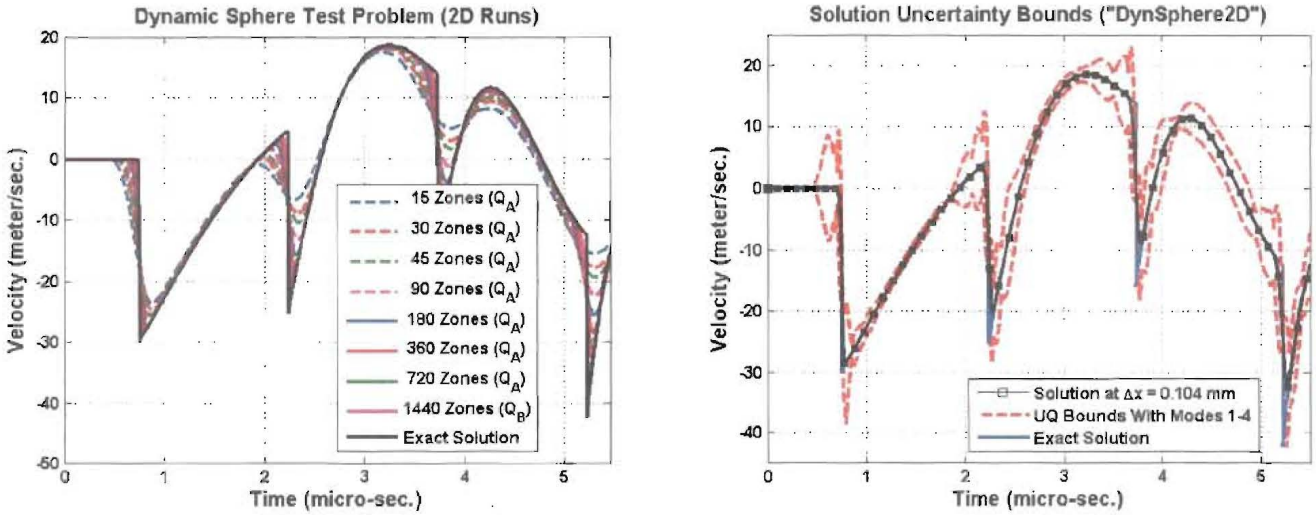

Figure 10. Solution uncertainty for a finite volume prediction of velocity curve [24].

Illustrations of solution uncertainty for entire curves are given in Figures 9 and 10. Figure 9 illustrates the prediction of a stress field for the finite element simulation of Hertz contact, see Reference [23]. The figure on the left depicts the coarse-mesh calculation where a rigid sphere applies pressure to an elastic region. The figure on the right shows stress values for the top row of finite elements (black, square symbols), the bounds of solution uncertainty (red, solid lines) and extrapolated solution (blue, solid line). The analysis indicates that, as we saw for Burgers equation, the bounds of solution uncertainty encompass the extrapolated solution.

Figure 10 conveys a similar message for the prediction of velocity field for a finite volume calculation, see Reference [24]. The problem is the propagation of a pressure front between the inner and outer surfaces of a solid sphere. The figure on the left compares the exact solution of the continuous equations (black, solid line) to several hydro-dynamics calculations performed with different mesh sizes and artificial viscosity settings. The figure on the right indicates that the bounds of solution uncertainty for a run performed at $\Delta x = 0.104$ mm include the exact solution except, as expected, at the very tip of sharp discontinuities.

7. CONCLUSION

The main theme of this publication is that discrete solutions computed by a computer code introduce ***numerical uncertainty***, which is one of many types of uncertainty that one needs to manage in physics and engineering applications. Other sources include experimental variability that introduces uncertainty on measurements, parametric variability that generates uncertainty on predictions, algorithmic selections and other assumptions needed to develop and implement models in computer simulations. Numerical uncertainty results from discretization. It should be quantified and accounted for in the overall uncertainty *"budget"* of the simulation; not doing so would introduce the risk of missing a significant uncertainty and jeopardizing decision-making.

To understand where numerical uncertainty comes from, the concepts of modified equation, consistency, convergence and truncation error are overviewed. The current state-of-the-practice of code and solution verification is discussed. A simple proposal is made to estimate bounds of solution uncertainty if the exact solution of the continuous equations being solved is unknown, which is always what happens in practice. This is when solution ***error*** becomes an ***uncertainty***. Several examples from finite element and hydro-dynamics calculations are given to illustrate the influence that meshing can exercise on the quality of discrete solutions and demonstrate the quantification of solution uncertainty.

On-going and future work includes addressing many small-yet-important details that arise during the implementation of solution uncertainty bounds. One such detail is how to perform a mapping of solution fields obtained from different meshes on a common, *"reference"* grid so that solution errors can be meaningfully estimated. Another track of research and development that shows promise is to perfect our understanding of the *"finite scale equations"* of a numerical method, as proposed in References [4-5], to implement better algorithms and enhance the rigor of code and solution verification analyses.

ACKNOWLEDGEMENTS

The author acknowledges the continued support at Los Alamos National Laboratory (LANL) of the Advanced Scientific Computing (ASC) program for Verification and Validation lead by Mark Anderson. The assistance that Randy Bos and Gregory Hutchens provided to perform the simulation runs is greatly appreciated. LANL is operated by the Los Alamos National Security, LLC for the National Nuclear Security Administration of the U.S. Department of Energy under contract DE-AC52-06NA25396.

REFERENCES

[1] Manson, G., Worden, K., **Proceedings of 1st International Conference on Uncertainty in Structural Dynamics**, University of Sheffield, United Kingdom, June 11-13, 2007.

[2] Ainsworth, M., Oden, J.T., **A Posterior Error Estimation in Finite Element Analysis**, *Wiley Inter-science Series in Pure and Applied Mathematics, John Wiley & Sons, Inc.*, New York City, New York, 2000, pp. 16-23.

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

[4] Margolin, L.G., Shashkov, M., "Finite Volume Methods and the Equations of Finite Scale: A Mimetic Approach," *International Journal for Numerical Methods in Fluids*, Vol. 56, 2008, pp. 991-1002.

[5] Margolin, L.G., "Finite Scale Equations for Compressible Fluid Flow," *Proceedings of the Physical Transactions of the Royal Society*, 2008, submitted.

[6] Kamm, J.R., Rider, W.J., Brock, J.S., "Combined Space and Time Convergence Analyses of a Compressible Flow Algorithm," *16th AIAA Computational Fluid Dynamics Conference*, Orlando, Florida, July 2003. (Los Alamos report LA-UR-03-2628.)

[7] Hemez, F.M., Brock, J.S., Kamm, J.R., "Non-linear Error Ansatz Models in Space and Time for Solution Verification," *8th AIAA Non-deterministic Approaches Conference*, Newport, Rhode Island, May 1-4, 2006. (Los Alamos report LA-UR-06-3705.)

[8] Lax, L.D., "Weak Solutions of Non-linear Hyperbolic Equations and Their Numerical Computation," *Communications on Pure and Applied Mathematics*, Vol. 7, 1954, pp. 159-193.

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

[10] Roy, C., "Review of Code and Solution Verification Procedures for Computational Simulation," *Journal of Computational Physics*, Vol. 205, 2005, pp. 131-156.

[11] Knupp, P., Salari, K., **Verification of Computer Codes in Computational Science and Engineering**, CRC Press, Boca Raton, Florida, 2002.

[12] Hemez, F.M., Marcilhac, M., "Analysis of Numerical Solution Error and Uncertainty Using Statistical Effect Screening," *26th SEM International Modal Analysis Conference*, Orlando, Florida, February 4-7, 2008. (Los Alamos report LA-UR-07-7768.)

[13] Hemez, F.M., "Challenges to the State-of-the-practice of Solution Convergence Verification," *9th AIAA Non-deterministic Approaches Conference*, Waikiki (Oahu), Hawai'i, April 23-26, 2007. (Los Alamos report LA-UR-06-8078.)

[14] Noh, W.F., "Errors for Calculations of Strong Shocks Using an Artificial Viscosity and an Artificial Heat-flux," *Journal of Computational Physics*, Vol. 72, No. 1, 1987, pp. 78-120.

[15] Hemez, F.M., Hutchens, G.J., Bos, R.J., "Verification of Mesh Adaptation Used in Lagrangian Hydro-dynamics Calculations," *Technical Report of the Fiscal Year 2008 ASC Verification and Validation Project "Code Verification,"* Los Alamos National Laboratory, Los Alamos, New Mexico, September 2008. (Los Alamos report LA-UR-08-6011.)

[16] Roache, P.J., "Perspective: A Method for Uniform Reporting of Grid Refinement Studies," *ASME Journal of Fluids Engineering*, Vol. 116, September 1994, pp. 405-413.

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

[18] Stern, F., Wilson, R., Shao, J., "Quantitative V&V of Computational Fluid Dynamics (CFD) Simulations and Certification of CFD Codes with Examples," *2004 ICHMT International Symposium on Advances in Computational Heat Transfer*, Norway, April 19-24, 2004. (Paper CHT-04-V1.)

[19] Germain, P., Bader, R., "Unicité des Ecoulements avec Chocs dans la Mécanique de Burgers," *Office National d'Etudes et de Recherches Aéronautiques (ONERA)*, Paris, France, 1953, pp. 1-13.

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

[21] LeVeque, R.J., **Finite Volume Methods for Hyperbolic Problems**, Cambridge University Press, 2002.

[22] Hemez, F.M., "Functional Data Analysis of Solution Convergence," *Technical Report Contributed to the Fiscal Year 2007 ASC Code Verification Project*, Los Alamos National Laboratory, Los Alamos, New Mexico, August 2007. (Reference: LA-UR-07-5758.)

[23] Hemez, F.M., Marcilhac, M., "A Fresh Look at Mesh Refinement in Computational Physics and Engineering," *10th AIAA Non-Deterministic Approaches Conference*, Schaumburg, Illinois, April 7-10, 2008. (Los Alamos report LA-UR-08-1731.)

[24] Hemez, F.M., Brock, J.S., Kamm, J.R., "Verification of a Lagrangian Hydro-dynamics Code with the Dynamic Sphere Problem," *8th World Congress on Computational Mechanics*, Venice, Italy, June 30-July 5, 2008. (Los Alamos report LA-UR-08-4197.)