Chapter 5
A Brief Overview of Code and Solution Verification in Numerical Simulation

François Hemez

Abstract This manuscript is a brief overview of code and calculation verification in computational physics and engineering. Verification is an essential technology to assess the quality of discrete solutions obtained by running simulation codes that solve systems of ordinary or partial differential equations, such as the finite element and finite volume methods. Code verification assesses the extent to which a numerical method is implemented correctly, that is, without any programming mistake (or "bug") that would adversely affect the quality of computations. The centerpiece of code verification is the formulation of verification test problems that admit exact or manufactured solutions and which are used for comparison with approximate solutions obtained from the simulation software. Solution verification assesses the extent to which the discretization (in time, space, energy, modal basis, etc.) implemented to solve governing equations provides a sufficiently small level of truncation error. The keystone of solution verification is the practice of mesh refinement from which estimates of the (spatial) order of accuracy of the numerical method can be estimated. It is also possible to derive bounds of truncation error produced in the calculation. The discussion is presented in the context of Peter Lax's 1954 groundbreaking work on the convergence of discrete solutions. It is illustrated with a simple example of one-dimensional advection solver. *(Publication approved for unlimited, public release, LA-UR-16-24553, Unclassified.)*

Keywords Code verification • Solution verification • Solution convergence • Truncation error

5.1 Introduction

In computational engineering and physics, ordinary or partial differential equations that govern the evolution of state variables, such as pressure, temperature, velocity, or displacement, are discretized for implementation and resolution on finite-digitized arithmetic computer. The challenge of verification is to assess the extent to which approximate solutions of the discretized equations converge to the exact solution of the continuous equations. In addition to assessing how "closely" discrete solutions estimate the continuous solution, it is often important to verify that the observed rate-of-convergence matches the theoretical order of accuracy of the numerical method or solver. Simply speaking, verification is the first "V" of Verification and Validation (V&V) for predictions of numerical simulations [1].

Code and calculation (or solution) verification is defined as a *scientifically rigorous and quantitative process for assessing the mathematical consistency between continuum and discrete variants of partial differential equations* [2]. Verification involves comparing numerical solutions obtained on successively refined meshes (or grids) to an exact solution or "reference." The main difficulty is that the exact solution of continuous equations is not always known and available to define this reference. These exact solutions can be derived analytically only in a few cases that feature "smooth" dynamics, linearized operators, simple geometries, or combinations of the above. Well-known examples include the single degree-of-freedom harmonic oscillator, linear vibration of simple structural components, Poiseuille flow of 2D incompressible fluids, and the 1D Riemann problem for ideal gases. These are problems that have been extensively studied and for which exact or highly-accurate solutions of the continuous equations can be obtained. Exact solutions can also be produced with the technique of manufactured solutions [3]. The overall number of problems that offer closed-form solutions is, unfortunately, limited.

In general one talks of *code verification* when the solution of the continuous equations can be derived in closed form, which provides an exact reference to which the discrete solutions are compared. By "closed form," we mean either analytical formulae or simplified representations that can be accurately evaluated, e.g., reduction from a partial differential equation to an ordinary differential equation. If the system of equations and its initial and boundary conditions are complicated, then

F. Hemez (✉)
"X" Theoretical Design Division, Los Alamos National Laboratory, XTD-IDA, Mail Stop T087, Los Alamos, NM 87545, USA
e-mail: hemez@lanl.gov

R. Barthorpe et al. (eds.), *Model Validation and Uncertainty Quantification, Volume 3*,
Conference Proceedings of the Society for Experimental Mechanics Series, DOI 10.1007/978-3-319-54858-6_5

a continuous solution cannot be derived analytically and one then talks of *solution verification*. In this latter case no exact reference is available to calculate the solution error due to discretization, which makes verification quite challenging.

The discussion presented herein offers a brief overview of technology applicable to code and solution verification alike. The only difference between the two cases is the availability (for code verification) or not (for solution verification) of a reference solution. The dominant paradigm for code and solution verification is that the numerical method provides discrete solutions that converge to the (possibly unknown) continuous solution with a specific rate-of-convergence (or order of accuracy). For example, a finite element analysis featuring quadratic shape functions should yield discrete solutions that converge with a rate of $p = 2$ when the mesh is refined. Common practice is to generate several discrete solutions from successively refined meshes, estimate an extrapolation of the continuous solution, verify the rate-of-convergence, and estimate bounds of numerical uncertainty. Discussion is restricted to well-established techniques and does not address recent developments that might challenge boundaries of the current state-of-the-practice. For a partial account of such developments see, for example, Refs. [4–10].

Successes of verification include the development of a formalism to study the convergence of discrete solutions for a wide range of applications, from linear, non-dissipative, elliptic equations to non-linear, shocked, hyperbolic equations. It means that the convergence, for example, of solutions for the resonant frequencies of a modal analysis can be studied with the same tools as the convergence of solutions for a blast wave propagating in ideal gases. The failures of verification are embodied by the many restrictions imposed on the way an analysis is typically carried out. These include restricting the studies to scalar quantities, further restricting them to spatial-only convergence, and not always accounting for how discrete solutions are constructed by the numerical methods and the specific properties that result.

5.2 The Consistency and Convergence of Modified Equations

Numerical methods, such as finite element and finite volume methods, solve conservation laws that balance the rate-of-change in time of state variables, gradient of their fluxes, and source terms. Without loss of generality these contributions can be written in a one-dimensional, Cartesian coordinate system as:

$$\frac{\partial y^{Exact}(x;t)}{\partial t} + \frac{\partial \mathrm{F}\left(y^{Exact}(x;t)\right)}{\partial x} = S(x;t)\,, \tag{5.1}$$

where $y^{Exact}(x;\,t)$ is the exact solution, F(•) denotes the flux operator, and $S(x;\,t)$ is a source or forcing function that drives the dynamics of the system. These generic functions depend on space and time, labeled x and t, respectively. When solving systems of Eq. (5.1) with a numerical method, one seeks the best-possible approximation of the continuous, exact, and often unknown solution, y^{Exact}.

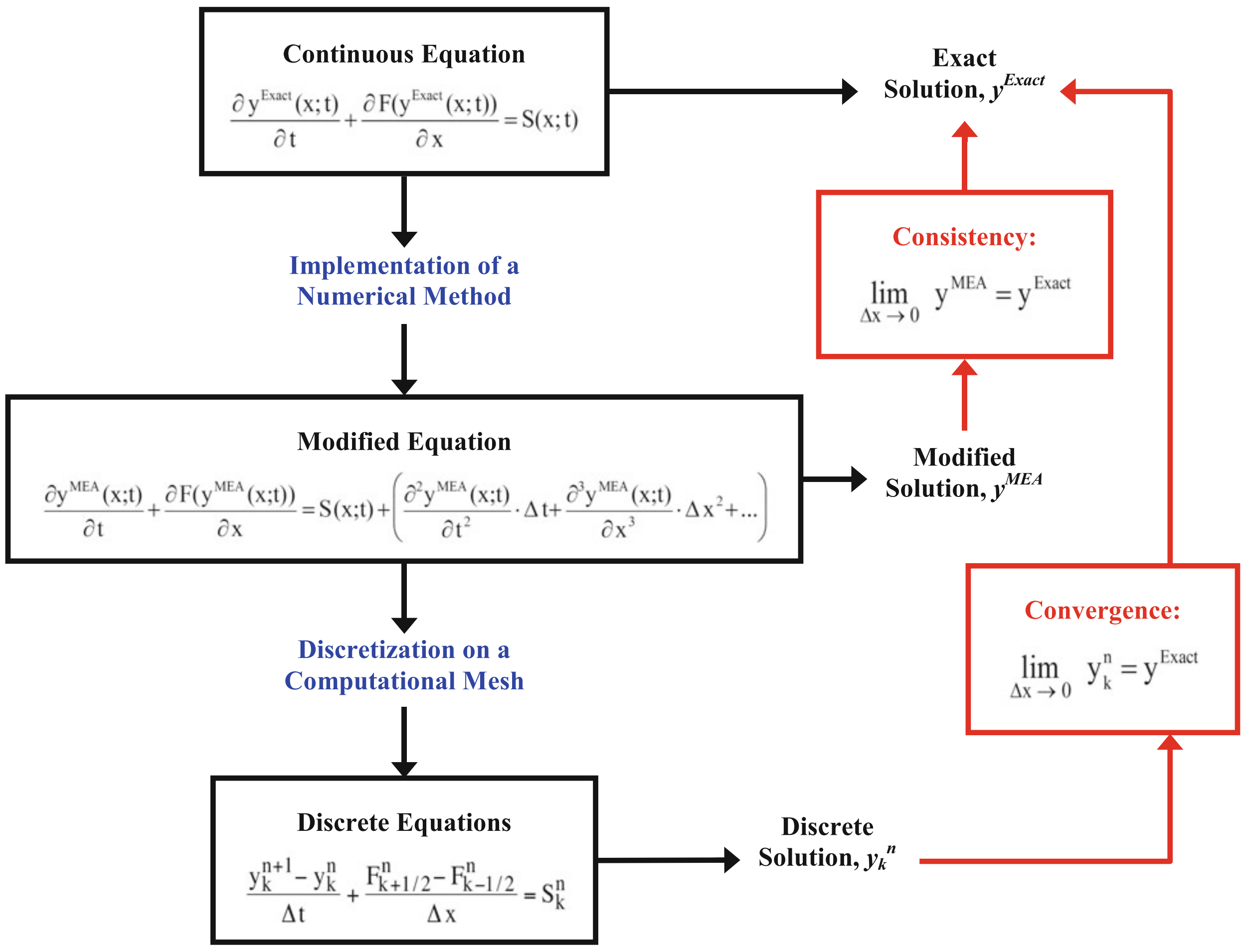
The numerical method discretizes the continuous Eq. (5.1) on a computational mesh to yield a discrete solution $y_k{}^n = y(k\cdot\Delta x;\ n\cdot\Delta t)$ where Δx and Δt are spatial and temporal resolutions. The approximation $y_k{}^n$ is obtained by solving a discretized system of equations that resembles, for example, something like:

$$\frac{y_k^{n+1} - y_k^n}{\Delta t} + \frac{F_{k+1/2}^n - F_{k-1/2}^n}{\Delta x} = S_k^n. \tag{5.2}$$

It is emphasized that Eq. (5.2) suggests a hypothetical discretization scheme (not necessarily the one used) where the temporal differentiation operator (∂•/∂t) is approximated by a forward Euler difference and the spatial differentiation operator (∂•/∂x) is approximated by central differences. The discrete values $y_k{}^n$, $F_k{}^n$ and $S_k{}^n$ can be obtained from finite difference, finite volume, or finite element approximations.

One observes that the discretized Eq. (5.2) seems "similar" to the original, continuous Eq. (5.1). The similarity, however, is misleading. Contrary to common belief the discrete solution $y_k{}^n$ does *not* approximate the continuous solution y^{Exact} of Eq. (5.1). Using the technique known as Modified Equation Analysis (MEA), see References [11, 12], it can be shown that the approximation $y_k{}^n$ converges to the solution of a *modified equation* that takes a form such as:

$$\begin{aligned}\frac{\partial y^{MEA}(x;t)}{\partial t} + \frac{\partial \mathrm{F}\left(y^{MEA}(x;t)\right)}{\partial x} &= S(x;t) \\ &+ \left(\frac{\partial^2 y^{MEA}(x;t)}{\partial t^2}\cdot\Delta t + \frac{\partial^3 y^{MEA}(x;t)}{\partial t^3}\cdot\Delta x^2 + \ldots\right).\end{aligned} \tag{5.3}$$



Continuous Equation
$\frac{\partial y^{Exact}(x;t)}{\partial t}+\frac{\partial F(y^{Exact}(x;t))}{\partial x}=S(x;t)$
Exact Solution, y^{Exact}
Implementation of a Numerical Method
Consistency:
$\lim_{\Delta x\to 0} y^{MEA}=y^{Exact}$
Modified Equation
$\frac{\partial y^{MEA}(x;t)}{\partial t}+\frac{\partial F(y^{MEA}(x;t))}{\partial x}=S(x;t)+\left(\frac{\partial^2 y^{MEA}(x;t)}{\partial t^2}\cdot\Delta t+\frac{\partial^3 y^{MEA}(x;t)}{\partial x^3}\cdot\Delta x^2+\ldots\right)$
Modified Solution, y^{MEA}
Convergence:
$\lim_{\Delta x\to 0} y_k^n=y^{Exact}$
Discretization on a Computational Mesh
Discrete Equations
$\frac{y_k^{n+1}-y_k^n}{\Delta t}+\frac{F_{k+1/2}^n-F_{k-1/2}^n}{\Delta x}=S_k^n$
Discrete Solution, y_k^n

Fig. 5.1 The concepts of consistency and convergence of numerical solutions

Note that this example is conceptual and the correct form of the modified equation depends on properties of the original Eq. (5.1) and numerical method (5.2) implemented for its resolution. What is important to the discussion is that the continuous solution y^{MEA} of the modified Eq. (5.3) is different from y^{Exact} as long as the spatial and temporal resolutions remain finite (as long as $\Delta x \neq 0$, $\Delta t \neq 0$). The Lax equivalence theorem of Ref. [13] details conditions under which the discrete approximation $y_k{}^n$ is *consistent* with, and *converges* to, the continuous solution y^{Exact}. Figure 5.1 illustrates these essential concepts.

Terms shown between parentheses in the right-hand side of the modified Eq. (5.3) represent an infinite series expansion that characterizes the *truncation error* of the numerical method or solver. Truncation is what explains the difference between the continuous solutions y^{MEA} and y^{Exact}. This is why it is essential to establish that the solution y^{MEA} of the modified Eq. (5.3) is consistent with the exact solution y^{Exact} of the original Eq. (5.1), as indicated in Fig. 5.1. Likewise one needs to understand the behavior of truncation error, that is, the extent to which the discrete solution $y_k{}^n$ converges to the exact solution y^{Exact}.

In addition to defining the concept of modified equation, Eq. (5.3) also illustrates the order of accuracy of a numerical method. The leading-order term of spatial truncation, for example, is proportional to Δx^2, which means that the solver is 2nd-order accurate. Performing a mesh refinement should yield a truncation error between discrete solutions $y_k{}^n$ and the exact-but-unknown solution y^{Exact} that converges at a quadratic rate. Likewise Eq. (5.3) suggests that the solver is 1st-order accurate in time since the leading-order term of temporal truncation is proportional to Δt. The next section discusses this asymptotic convergence.

5.3 The Regime of Asymptotic Convergence of Discrete Solutions

Because truncation error is often the main mechanism by which discrete solutions $y_k{}^n$ differ from the exact solution y^{Exact}, understanding its behavior is key to assess the numerical performance of simulation software. Verifying the quality of discrete solutions hinges on the *regime of asymptotic convergence*.

Different choices of discretization variables, such as Δx or Δt, induce different types and magnitudes of error. Plotting the solution error $||y^{Exact} - y_k{}^n||$ as a function of mesh size Δx gives a conceptual illustration of the main three regimes of the discretization. By definition the asymptotic regime is the region where truncation dominates the overall production of numerical error. Figure 5.2 provides a simplified illustration of these regimes. They are color-coded such that *red* is the regime where discretization is inappropriate, *green* denotes the regime of asymptotic convergence, and *grey* is where round-off error accumulates.

Going from right (larger values of Δx) to left (smaller values of Δx) in Fig. 5.2, the first domain shown in *red* is where the choice of element or cell size is not even appropriate to solve the discrete equations. This would be, for example, the case when elements are too coarse to resolve important geometrical features of a contact condition between components, or a numerical stability criterion is violated. Although it could be argued that discrete solutions should not be computed in this regime, the fact is undeniable that meshes analyzed in practical situations are often significantly under-resolved.

The second region shown in *green* is where truncation dominates the overall error: it is the regime of asymptotic convergence. Because truncation error dominates, the accuracy of a discrete solution y_k can be improved simply by performing the calculation with a smaller element size. We have just described the basic principle of conducting a mesh or grid refinement study.

Our conceptual illustration assumes that truncation error within the regime of asymptotic convergence is dominated by a single effect proportional to Δx^p, which appears as a straight line of slope p on the log-log representation of Fig. 5.2. The functional form of a typical modified equation, suggested in Eq. (5.3), motivates this assumption. It is also the reason why the behavior of truncation error is usually studied by formulating a simple model such as:

$$\varepsilon(\Delta x) = \left\| y^{Exact} - y_k^n(\Delta x) \right\| \approx \beta \cdot \Delta x^p + \text{H.O.T.}, \tag{5.4}$$

where $\varepsilon(\Delta x)$ denotes the difference, estimated in the sense of a user-defined norm $||\bullet||$, between the exact solution y^{Exact} of the continuous Eq. (5.1) and the discrete solution $y_k{}^n(\Delta x)$ obtained by executing the simulation code with mesh or grid size Δx. The pre-factor β represents a (constant) regression coefficient. The exponent p characterizes the rate at which the solution error decreases as the level of resolution is increased, that is, as $\Delta x \rightarrow 0$. It should match the theoretical order of

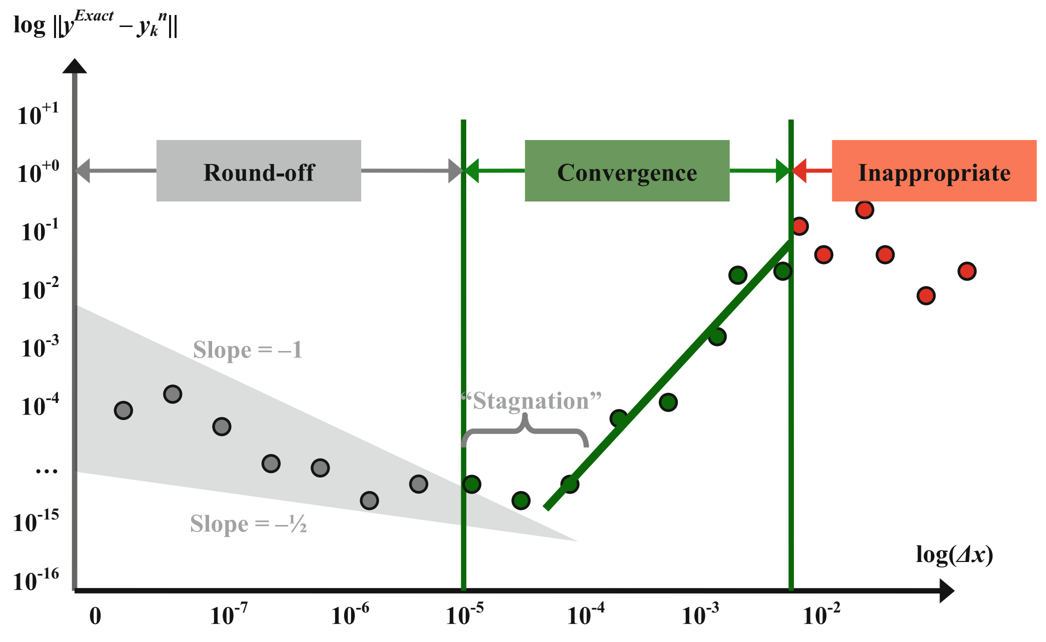


Fig. 5.2 The three regimes of solution error as a function of spatial discretization

accuracy of the numerical method. For example, a finite element calculation that uses linear shell elements should exhibit the rate of $p = 1$. Likewise the second-order accurate Gudonov scheme of a fluid dynamics solver should feature $p = 2$ for a laminar-like flow that does not develop a discontinuity (or shock).

The last region of Fig. 5.2 shown in *grey* is a limiting case for asymptotic convergence due to the fact that, with finite arithmetic implemented in a computer code, round-off errors eventually start to grow as $\Delta x \rightarrow 0$. Round-off could then accumulate to the point where it supplants truncation as the dominant mechanism that produces numerical error.

Understanding asymptotic convergence is important for two reasons. First, the formulae discussed in this manuscript, which provide the foundation of verification, are valid only within the regime of asymptotic convergence. Second, verifying that a discretization used in a calculation leads to a discrete solution in the asymptotic regime provides a strategy to reduce truncation error. If the error is too large for the application and needs to be reduced, performing the simulation with a smaller element or cell size automatically reduces it. This is, however, true only within the regime of asymptotic convergence.

5.4 State-of-the-Practice of Code and Solution Verification

This section summarizes the formalism and main equations used to study the convergence of discrete solutions. The discussion focuses on the state-of-the-practice in computational engineering and physics without paying tribute to some of the more advanced techniques. References [5, 9, 10] to list only a few, discuss issues that extend beyond the limited scope of this section.

The distinction needs to be emphasized between *code verification*, where the exact solution y^{Exact} of the continuous Eq. (5.1) is known analytically, or provided by an approximate yet highly accurate solution procedure, and *solution verification* where y^{Exact} is unknown. In the case of code verification, the error $\varepsilon(\Delta x)$ in the left-hand side of Eq. (5.4) can be computed given knowledge of the exact solution y^{Exact} and a discrete solution $y_k{}^n(\Delta x)$ obtained with the discretization size Δx. These exact and discrete solutions can be any scalar quantities, curves, or multi-dimensional fields. The unknowns of Eq. (5.4) are the parameters (β; p). Two discrete solutions, one obtained with a "coarse" mesh of size Δx_C and another one obtained with a "fine" mesh of size Δx_F, where $\Delta x_C = R \cdot \Delta x_F$ (and $R > 1$), suffice to estimate β and p.

We now proceed to discuss the general case of solution verification where the difficulty is that the exact solution y^{Exact} of the continuous Eq. (5.1) is unknown. The challenge is that the procedure described in the previous paragraph cannot be implemented since the error $\varepsilon(\Delta x)$ cannot be explicitly computed. The starting point is to postulate an equation, referred to as an Ansatz model of numerical error, which describes how the error behaves in the regime of asymptotic convergence. The state-of-the-practice is to specialize Eq. (5.4) to scalar quantities such that is can be written as:

$$\varepsilon\left(\Delta x\right) = y^{Reference} - y_k^n\left(\Delta x\right) \approx \beta \cdot \Delta x^p, \tag{5.5}$$

where the unknown is the triplet of quantities ($y^{Reference}$; β; p). The main difference with code verification defined in Eq. (5.4) is that the exact-but-unknown solution y^{Exact} is replaced by a reference $y^{Reference}$ that becomes a third unknown of the Ansatz model. Equation (5.5) also assumes that, first, the convergence of discrete solutions $y_k{}^n(\Delta x)$ towards the reference solution is monotonic and, second, the higher-order terms of the modified equation can be neglected. References [9, 10] address the case of non-monotonic convergence.

Because the formulation (5.5) features three unknowns, a minimum of three equations is required to solve it. These are provided by discrete solutions obtained from a coarse-mesh calculation (with element or cell size Δx_C), a medium-mesh calculation (Δx_M), and a fine-mesh calculation (Δx_F). These are written as:

$$\begin{aligned} y^{Reference} - y_k^n\left(\Delta x_C\right) &= \beta \cdot \Delta x_C^p \\ y^{Reference} - y_k^n\left(\Delta x_M\right) &= \beta \cdot \Delta x_M^p . \\ y^{Reference} - y_k^n\left(\Delta x_F\right) &= \beta \cdot \Delta x_F^p \end{aligned} \tag{5.6}$$

The system of Eq. (5.6) can be solved for the triplet of unknowns ($y^{Reference}$; β; p) by combining the equations to eliminate two of the three unknowns. The value of the rate-of-convergence p, for example, is obtained by solving the following nonlinear equation:

$$p \cdot \log\left(R_{MF}\right) + \log\left(1 - R_{CM}^p\right) - \log\left(1 - R_{MF}^p\right) = \log\left(\frac{y_k^n\left(\Delta x_M\right) - y_k^n\left(\Delta x_C\right)}{y_k^n\left(\Delta x_F\right) - y_k^n\left(\Delta x_M\right)}\right), \tag{5.7}$$

where R_{CM} and R_{MF} are the successive mesh refinement ratios defined as:

$$R_{CM} = \frac{\Delta x_C}{\Delta x_M} \quad \text{and} \quad R_{MF} = \frac{\Delta x_M}{\Delta x_F}. \tag{5.8}$$

There is no closed-form solution to the nonlinear Eq. (5.7) when the refinement ratios $\mathrm{R_{CM}}$ and $\mathrm{R_{MF}}$ are different. The value of exponent p must be obtained through numerical optimization. With a constant mesh refinement ratio, which means that $R_{CM} = R_{MF} = R$, however, Eq. (5.7) is solved as:

$$p = \frac{\log\left(\dfrac{y_k^n(\Delta x_M) - y_k^n(\Delta x_C)}{y_k^n(\Delta x_F) - y_k^n(\Delta x_M)}\right)}{\log(R)}. \tag{5.9}$$

The next step is to extrapolate the discrete solutions $y_k{}^n(\Delta x_C)$, $y_k{}^n(\Delta x_M)$ and $y_k{}^n(\Delta x_F)$ to a reference $y^{Reference}$ that approximates the exact-but-unknown solution y^{Exact}. This is referred to as Richardson's extrapolation. Any combination of two meshes (coarse-medium or medium-fine) provides the same value for $y^{Reference}$:

$$y^{Reference} = y_k^n(\Delta x_F) + \frac{y_k^n(\Delta x_F) - y_k^n(\Delta x_M)}{R^p - 1} = y_k^n(\Delta x_M) + \frac{y_k^n(\Delta x_M) - y_k^n(\Delta x_C)}{R^p - 1}, \tag{5.10}$$

where the convergence rate p is the value obtained from Eq. (5.9). The regression coefficient β can be back-calculated from any one of Eq. (5.6):

$$\beta = \frac{y_k^n(\Delta x_C) - y_k^n(\Delta x_M)}{\Delta x_M^p \cdot (R^p - 1)} = \frac{y_k^n(\Delta x_M) - y_k^n(\Delta x_F)}{\Delta x_F^p \cdot (R^p - 1)} = \frac{y_k^n(\Delta x_C) - y_k^n(\Delta x_F)}{\Delta x_F^p \cdot (R^{2p} - 1)}. \tag{5.11}$$

In summary, Eqs. (5.7)–(5.11) provide expressions for the triplet of unknowns ($y^{Reference}$; β; p) in the case of either a variable or constant mesh refinement ratio. This formalism is applicable to code and solution verification alike, with the difference that the former case assumes knowledge of the exact solution y^{Exact}, which reduces the unknowns of the Ansatz model to the pair (β; p). Clearly this analysis technique is appropriate only to the extent that assumptions upon which it relies are satisfied. There are four essential assumptions summarized to conclude the discussion. (1) The mesh sizes used (Δx_C, Δx_M and Δx_F) provide discrete solutions located in the regime of asymptotic convergence. (2) The analysis is restricted to scalar quantities. (3) Because it is unknown, the exact solution y^{Exact} of the continuous equations is replaced by a to-be-estimated reference $y^{Reference}$. (4) Convergence of the discrete solutions $y_k{}^n(\Delta x)$ is monotonic.

5.5 The Bounds of Numerical Uncertainty

When the exact solution y^{Exact} of a system of partial or ordinary differential equations is unknown, the error $|y^{Exact} - y_k{}^n(\Delta x)|$ or $||y^{Exact} - y_k{}^n(\Delta x)||$ with which a discrete solution $y_k{}^n(\Delta x)$ is calculated becomes an *uncertainty*. This uncertainty cannot be explicitly computed since the exact solution is unknown and the approximation provided by the reference solution $y^{Reference}$ is itself uncertain. The best that one can achieve, therefore, is an upper bound $U(\Delta x)$ of truncation error such that:

$$\left| \frac{y^{Exact} - y_k^n(\Delta x)}{y_k^n(\Delta x)} \right| \le U(\Delta x). \tag{5.12}$$

References [14, 15] derive such an upper bound and illustrate how it can be used to select an appropriate mesh size for the prediction of resonant dynamics of a wind turbine blade. The result is given as:

$$U(\Delta x) = \frac{1}{R^p - 1} \cdot \left| \frac{y_k^n(\Delta x) - y_k^n(R \cdot \Delta x)}{y_k^n(\Delta x)} \right|, \tag{5.13}$$

where R denotes the refinement ratio used between two successive calculations. By analogy with previous notation, the mesh size $R{\cdot}\Delta x$ represents a coarse resolution (previously denoted as Δx_C) and Eq. (5.13) defines the upper bound of truncation error for a fine-resolution (previously denoted as Δx_M) prediction.

The significance of the upper bound (5.13) is to indicate where the exact-but-unknown solution y^{Exact} might be located relative to the prediction $y_k{}^n(\Delta x)$ obtained at mesh resolution Δx. It is analogous to the Grid Convergence Index (GCI) proposed by Patrick Roache in 1994 to report the results of grid convergence studies in computational fluid dynamics [16]. The definition (5.13) does not, however, suffer from the disadvantage of having to select a somewhat arbitrary "factor-of-safety" F_S that appears in the GCI:

$$GCI\,(\Delta x) = \frac{F_S}{R^p - 1} \cdot \left| \frac{y_k^n\,(\Delta x) - y_k^n\,(R \cdot \Delta x)}{y_k^n\,(\Delta x)} \right|, \tag{5.14}$$

where the value of F_S depends on the application, characteristics of the code used and smoothness of the solution. Values $F_S = 1.25$ and 3.0 have been proposed for various applications in computational fluid dynamics and solid mechanics [2]. Some practitioners advocate to make the factor-of-safety variable to account for better or worse-than-expected convergence. Reference [17], for example, offers a definition that increases the upper bound if the observed rate-of-convergence is worse than anticipated:

$$F_S = \frac{R^{p^{Theory}} - 1}{R^p - 1}, \tag{5.15}$$

where p^{Theory} and p denote the theoretical and observed (in Eq. (5.7) or (5.9)) rates-of-convergence.

A small value of the upper bound $U(\Delta x)$, or the GCI, indicates that, first, the discrete solution obtained at mesh resolution Δx is converging to the exact-but-unknown solution y^{Exact} and, second, Richardson's extrapolation $y^{Reference}$ is a reasonably accurate estimation of y^{Exact}. What one considers "small enough" is subjective, usually, $U(\Delta x) = 10\%$ or less.

5.6 An Application of Solution Verification to a One-dimensional Advection Solver

An example is discussed to illustrate how solution verification is conducted in practice. Simplicity of our example should not obscure that the procedure would be similar for a real-world application in computational physics or engineering. Reference [14], for example, shows an application to wind turbine blade vibration using a finite element model.

The example is an advection equation solved in one-dimensional, Cartesian coordinates with a diffusion term added in the right-hand side:

$$\frac{\partial \psi\,(x;t)}{\partial t} + \alpha \cdot \frac{\partial \psi\,(x;t)}{\partial x} = \beta \cdot \frac{\partial^2 \psi\,(x;t)}{\partial x^2} + S\,(x;t)\,, \tag{5.16}$$

where the computational domain is defined as $-1 \le x \le +1$. The initial (at $t = 0$) and boundary (at $x = \pm 1$) conditions are defined below. With the absence of a source function, that is, $S(x;t) = 0$ in Eq. (5.16), the waveform $\Psi(x;t)$ is simply advected (or translated) at constant velocity α across the computational domain. The coefficient β controls the amount of diffusion added to the solution. The solution $\Psi(x;t)$ is continuous, meaning that no discontinuity (or shock) or instability develops as time progresses.

An upwind scheme, which is 1st-order accurate, is selected to discretize the advection operator while central differences, which are 2nd-order accurate, are used for the diffusion term. These choices are written as:

$$\begin{gathered} \frac{\partial \psi\,(x;t)}{\partial t} \approx \frac{1}{\Delta t} \cdot \left(\psi_k^{n+1} - \psi_k^n\right), \quad \frac{\partial \psi\,(x;t)}{\partial x} \approx \frac{1}{\Delta x} \cdot \left(\psi_{k+1}^n - \psi_k^n\right). \\ \frac{\partial^2 \psi\,(x;t)}{\partial x^2} \approx \frac{1}{\Delta x^2} \cdot \left(\psi_{k+1}^n - 2\psi_k^n + \psi_{k-1}^n\right) \end{gathered} \tag{5.17}$$

The resulting finite volume scheme is fully explicit and implemented in a Matlab™ simulation code as:

$$\psi_k^{n+1} = \psi_k^n + \frac{\alpha \cdot \Delta t}{\Delta x} \cdot \left(\psi_{k+1}^n - \psi_k^n\right) + \frac{\beta \cdot \Delta t}{\Delta x^2} \cdot \left(\psi_{k+1}^n - 2\psi_k^n + \psi_{k-1}^n\right) + S_k^n, \tag{5.18}$$

where $\Psi_k{}^n$ denotes the discretized zone-averaged solution at location $x = k \cdot \Delta x$ and time $t = n \cdot \Delta t$:

$$\psi_k^n = \int_{x=\left(k-\frac{1}{2}\right)\Delta x}^{x=\left(k+\frac{1}{2}\right)\Delta x} \psi\,(x; n \cdot \Delta t) \cdot dx. \tag{5.19}$$

Modified equation analysis provides the continuous equation that the discrete solution $\Psi_k{}^n$ attempts to approximate. The derivations (not shown) indicate that it consists of the original Eq. (5.16) augmented with an infinite number of terms that represent the truncation error introduced by the particular choice of discretization scheme (5.17). The leading-order terms of the modified equation are:

$$\begin{aligned}\frac{\partial\psi}{\partial t} + \alpha \cdot \frac{\partial\psi}{\partial x} = \beta \cdot \frac{\partial^2\psi}{\partial x^2} + S + \frac{\alpha\Delta x}{2} \cdot (1 - CFL) \cdot \frac{\partial^2\psi}{\partial x^2} - \frac{\alpha\Delta x^2}{6} \cdot \left(1 - CFL^2\right) \cdot \frac{\partial^3\psi}{\partial x^3}, \\ + \frac{\beta\Delta t}{2} \cdot \left(\alpha\frac{\partial^3\psi}{\partial x^3} - \beta\frac{\partial^4\psi}{\partial x^4}\right) - \frac{\beta\Delta t^2}{3} \cdot \left(\alpha^2\frac{\partial^4\psi}{\partial x^4} - \alpha\frac{\partial^5\psi}{\partial x^5} + \frac{\beta^2}{2}\frac{\partial^6\psi}{\partial x^6}\right) + \dots\end{aligned} \tag{5.20}$$

with the Courant number defined as $CFL = \alpha \cdot \Delta t / \Delta x$. The fully explicit numerical scheme of Eq. (5.18) is stable only if the time-space discretization (Δt; Δx) satisfies $CFL < 1$. The modified equation clearly indicates that the proposed numerical method is 1st-order accurate in both time and space since the leading terms of truncation error are proportional to Δt (for the temporal error) and Δx (for the spatial error).

We now assume that modified equation analysis cannot be carried out. This is usually the case for real-world applications that feature multiple dimensions, large numbers of state variables, coupled mechanics or physics, and nonlinear governing equations, which makes it intractable to derive the modified equation. When confronted with this situation, the behavior of truncation error is studied through mesh refinement.

The exact solution of the homogeneous variant of Eq. (5.16), for which $S(x;t) = 0$, is:

$$\psi^{Exact}\,(x; t) = \psi_0 \cdot e^{+\lambda t} \cdot \sin\left(kx + \omega t\right), \tag{5.21}$$

where the constants are defined as $\omega = -\alpha \cdot k$ and $\lambda = -\beta \cdot k^2$. In the numerical application discussed next, the coefficients are defined as $\alpha = 0.2$, $\beta = 0.002$, $\Psi_o = 1$ and $k = 15$. In addition, the initial and boundary conditions are defined to guarantee that $S(x; t)$ remains equal to zero at all locations and instants with:

$$\begin{aligned}\psi^{Exact}\,(x; t = 0) &= \psi_0 \cdot \sin(kx) \\ \psi^{Exact}\,(x = -1; t) &= \psi_0 \cdot e^{-\beta k^2 t} \cdot \sin\left(-k\left(\alpha t + 1\right)\right), \\ \psi^{Exact}\,(x = +1; t) &= \psi_0 \cdot e^{-\beta k^2 t} \cdot \sin\left(-k\left(\alpha t - 1\right)\right).\end{aligned} \tag{5.22}$$

Figure 5.3 illustrates exact solutions $\Psi^{Exact}(x;t)$ at times $t = 0$ (initial condition) and $t = 1$. The two physical effects can be observed with, first, advection that translates the solution across the domain (from the red curve to the blue curve) and, second, diffusion that removes energy from the solution, which tends to "spread" the waveform spatially and reduces its amplitude.

Table 5.1 defines six discrete solutions used to execute a mesh refinement study. The refinement ratio is uniform and equal to $R = 2$. Settings selected for these runs satisfy the stability criterion ($CFL < 1$), even for the most refined discretization with 3040 zones. Figure 5.4 illustrates the discrete solutions (color lines) and compares them to the exact solution (black dashed line) at $t = 1$. The figure clearly indicates that the discrete solutions converge to the exact solution (5.21) without, however, revealing at which rate.

Formal verification is carried out next. Code verification could be performed using the entire solution field by studying the behavior of truncation error $||\Psi^{Exact} - \Psi_k{}^n||$ since the exact solution is analytically known as shown in Eqs. (5.21) and (5.22). It is not, however, what is illustrated here. The exact solution is *not* used and solution verification is performed instead. To simplify the discussion, it is applied to peak values extracted from the discrete solutions at the final simulation time of $t = 1$, that is:

$$_{Max}\psi_k^n = \max_{-1 \le x \le +1} \psi_k^n\,(x; 1)\,. \tag{5.23}$$

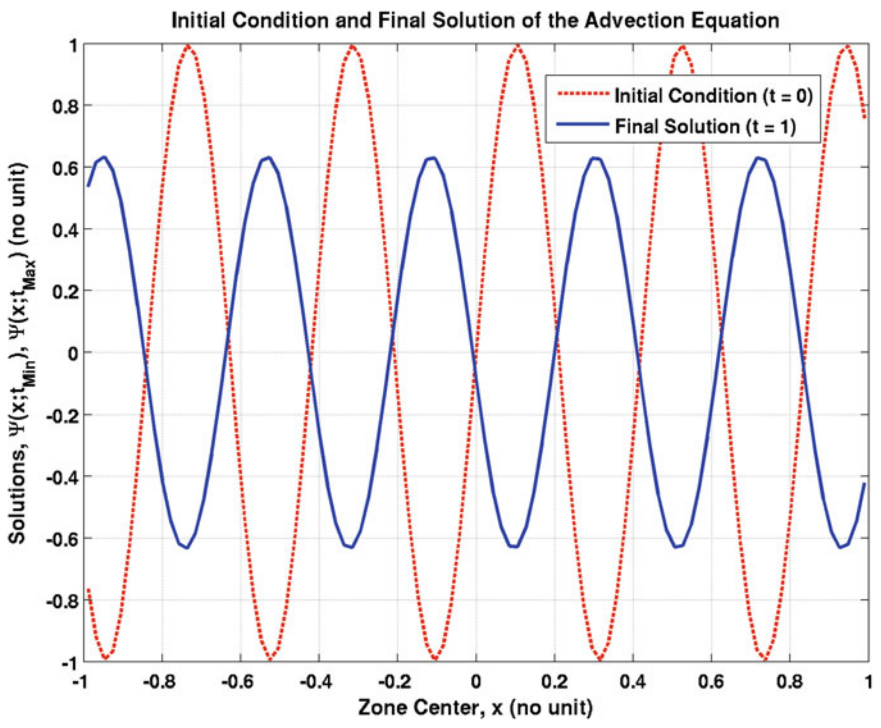


Fig. 5.3 Exact solutions Ψ^{Exact} at times $t = 0$ (*red* curve) and $t = 1$ (*blue* curve)

Table 5.1 Definition of settings used to generate six discrete solutions $\Psi_k{}^n$

Solution	N_{Zone}	Δx	Δt	CFL
1	95	21.052×10^{-3}	10^{-4}	9.5×10^{-4}
2	190	10.526×10^{-3}	10^{-4}	19.0×10^{-4}
3	380	5.263×10^{-3}	10^{-4}	38.0×10^{-4}
4	760	2.632×10^{-3}	10^{-4}	76.0×10^{-4}
5	1520	1.316×10^{-3}	10^{-4}	152.0×10^{-4}
6	3040	0.658×10^{-3}	10^{-4}	304.0×10^{-4}

The objectives of the analysis are to, first, verify if the peak values pictured in Fig. 5.4 converge with a 1st-order rate when the computational mesh is refined, as they should according to the modified Eq. (5.20); and, second, quantify the solution uncertainty at any level of mesh resolution.

Even though it would be possible to use triplets of discrete solutions to estimate the rate-of-convergence using Eq. (5.9) with $R = 2$, the behavior of truncation error is studied by simultaneously processing the six discrete solutions and solving Eq. (5.7) using a nonlinear optimization solver. It gives:

$$\left|{}_{Max}\psi^{Reference} - {}_{Max}\psi_k^n\,(\Delta x)\right| = 3.454 \cdot \Delta x^{0.998}, \tag{5.24}$$

with ${}_{Max}\Psi^{Reference} = 0.634$. The observed convergence rate of $p = 0.998$ clearly indicates 1st-order spatial convergence, which matches expectation given the theoretical properties of the numerical solver (5.18).

Table 5.2 lists the bounds of solution uncertainty $U(\Delta x)$ for the four finest levels of resolution. The fourth column gives the upper bounds as percentages of discrete solutions, as defined in Eq. (5.13), and the fifth column gives them in absolute magnitudes, that is, ${}_{Max}\Psi_k{}^n(\Delta x)\cdot U(\Delta x)$. These discrete solutions, all obtained with a minimum of 380 computational zones, feature less than 10% numerical uncertainty due to truncation error. Another factor, that Table 5.2 does not show, is time-to-solution. If an overall level of 10% error is admissible to predict the peak value, one might decide to pursue simulations using the 380-zone discretization because its solution can be computed faster than using a finer level of mesh resolution.

Figure 5.5 summarizes the results by illustrating the discrete solutions (blue squares), which are the peak values ${}_{Max}\Psi_k{}^n(\Delta x)$ obtained at different resolutions Δx, and their bounds of uncertainty (vertical green lines), which are the upper bounds $U(\Delta x)$ that result from mesh-induced truncation error. Note that the figure uses a log-log scale on its horizontal (mesh sizes) and vertical (solutions) axes. The red dashed line represents the 1st-order trend with which discrete solutions converge.

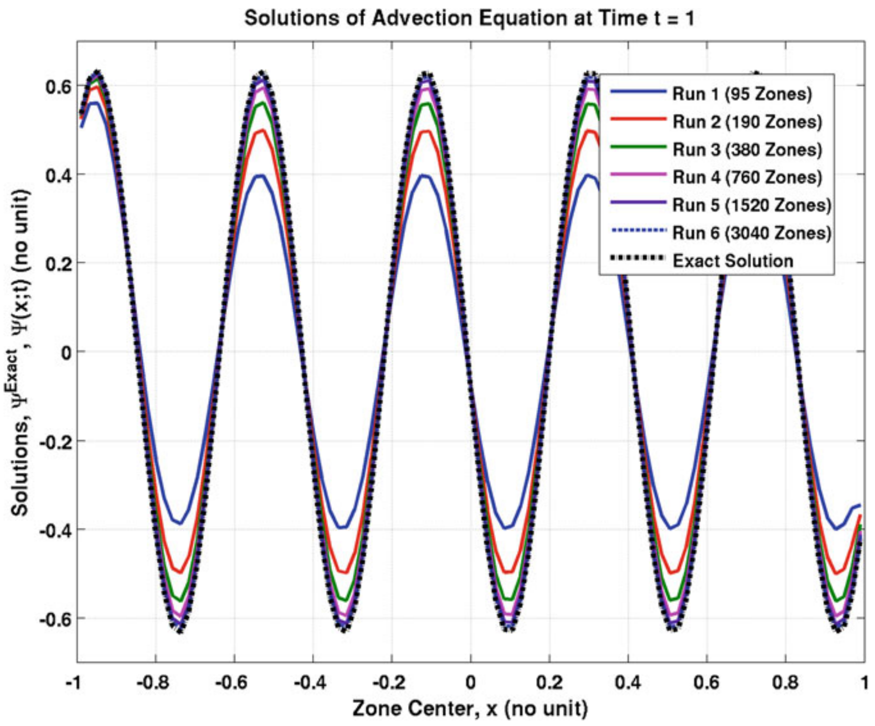


Fig. 5.4 Comparison of discrete $\Psi_k{}^n$ (*color lines*) and exact Ψ^{Exact} (*black dashed line*) solutions

Table 5.2 Bounds of solution uncertainty $U(\Delta x)$ at different mesh resolutions

Solution	N_{Zone}	Δx	$U(\Delta x)$	$_{Max}\Psi_k{}^n(\Delta x)\bullet U(\Delta x)$
3	380	5.263×10^{-3}	9.12%	0.0561
4	760	2.632×10^{-3}	4.24%	0.0265
5	1520	1.316×10^{-3}	2.09%	0.0132
6	3040	0.658×10^{-3}	1.05%	0.0066

The horizontal black dashed line indicates the extrapolated solution $_{Max}\Psi^{\text{Reference}}$, which is the best estimate of the value that the numerical solver would converge to if a calculation could be performed at infinite resolution, that is, as $\Delta x \rightarrow 0$.

5.7 Conclusion

Verifying that a simulation code is free of programming mistakes, and the numerical methods that it implements deliver convergence properties which match theory, are essential activities to establish the quality of numerical solutions. Simply speaking, it is the first "V" of the Verification and Validation of numerical simulations. The discipline of code and solution verification in computational engineering and physics develops tools to assess the sources and severity of discretization and numerical errors. The discussion offered in this manuscript outlines simple procedures to evaluate the regime of asymptotic convergence, estimate a rate-of-convergence based on mesh (or grid) refinement, extrapolate discrete solutions, and develop bounds of numerical uncertainty due to discretization-induced truncation error.

Much of verification, however, remains incomplete. A non-exhaustive list of topics that warrant further research includes: extending the state-of-the-practice of verification to non-scalar quantities (curves, images, multiple-dimensional fields), developing technology to study the coupling between temporal and spatial discretization, defining a "reference" mesh for the estimation of solution error, and developing methods to verify the numerical quality of calculations which feature adaptive mesh refinement. It is our hope that computational engineers and physicists, while increasingly relying on modeling and simulation, will recognize the importance that numerical issues play in model validation and become advocates for best practices in the discipline of code and solution verification.

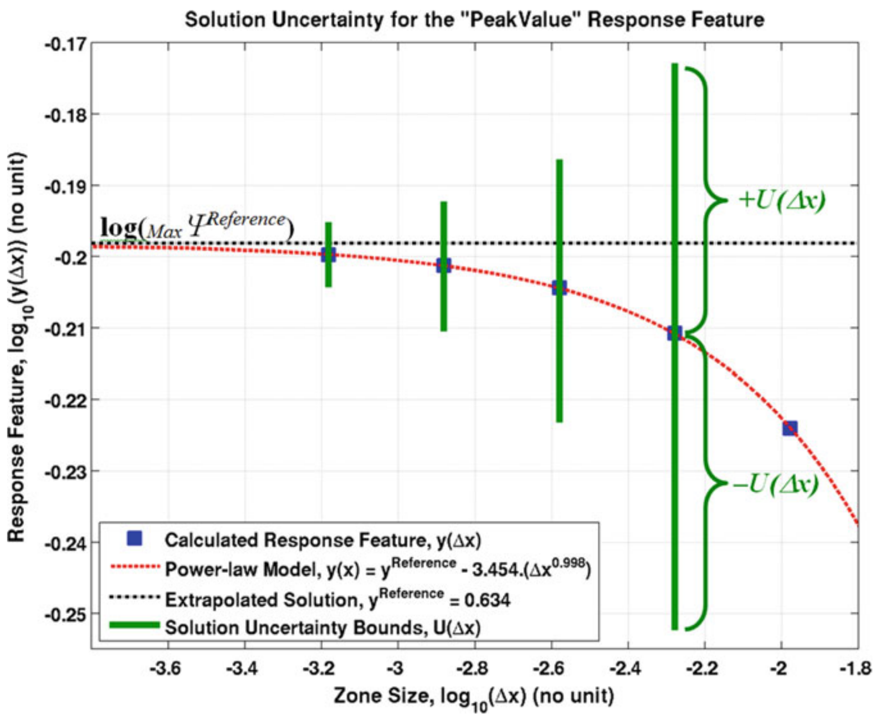


Fig. 5.5 Convergence of discrete solutions ${}_{Max}\Psi_k{}^n$ and their bounds of truncation error $U(\Delta x)$

Acknowledgments The author gratefully acknowledges the support of the Advanced Certification Campaign at the Los Alamos National Laboratory (LANL). LANL is operated by Los Alamos National Security, L.L.C., under contract DE-AC52-06NA25396 on behalf of the National Nuclear Security Administration of the U.S. Department of Energy.

References

1. Hemez, F.M., Doebling, S.W., Anderson, M.C.: A brief tutorial on verification and validation. In: 22nd SEM International Modal Analysis Conference, Dearborn, Michigan (2004)
2. Roache, P.J.: Verification in Computational Science and Engineering. Hermosa Publishers, Albuquerque, NM (1998)
3. Salari, K., Knupp, P.: Code verification by the method of manufactured solutions, Technical Report SAND-2000-1444, Sandia National Laboratories, Albuquerque, NM (2000)
4. Kamm, J.R., Rider, W.J., Brock, J.S.: Consistent metrics for code verification, Technical Report LA-UR-02-3794, Los Alamos National Laboratory, Los Alamos, NM (2002)
5. Kamm, J.R., Rider, W.J., Brock, J.S.: Combined space and time convergence analyses of a compressible flow algorithm. In: 16th AIAA Computational Fluid Dynamics Conference, Orlando, FL (2003)
6. Knoll, D.A., Chacon, L., Margolin, L.G., Mousseau, V.A.: On balanced approximations for time integration of multiple time scale systems. J. Comput. Phys. **185**(2), 583–611 (2003)
7. Buechler, M., McCarty, A., Reding, D., Maupin, R.D.: Explicit finite element code verification problems. In: 22nd SEM International Modal Analysis Conference, Dearborn, MI (2004)
8. Li, S., Rider, W.J., Shashkov, M.J.: Two-dimensional convergence study for problems with exact solution: uniform and adaptive grids, Technical Report LA-UR-05-7985, Los Alamos National Laboratory, Los Alamos, NM (2005)
9. Smitherman, D.P., Kamm, J.R., Brock, J.S.: Calculation verification: point-wise estimation of solutions and their method-associated numerical error. J. Aerosp. Comput. Inf. Commun. **4**, 676–692 (2007)
10. Hemez, F.M., Brock, J.S., Kamm, J.R.: Nonlinear error ansatz models in space and time for solution verification. In: 1st Non-deterministic Approaches Conference and 47th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, Newport, RI (2006)
11. Warming, R., Hyett, B.: The modified equation approach to the stability and accuracy analysis of finite difference methods. J. Comput. Phys. **14**, 159–179 (1974)
12. LeVeque, R.J.: Numerical Methods for Conservation Laws. Birkhauser-Verlag Publishers, Basel (1990)
13. Lax, P.D., Richtmyer, R.D.: Survey of the stability of linear finite difference equations. Commun. Pure Appl. Math. **9**, 267–293 (1956)
14. Mollineaux, M.G., Van Buren, K.L., Hemez, F.M., Atamturktur, S.: Simulating the dynamics of wind turbine blades: part I, model development and verification. Wind Energy. **16**, 694–710 (2013)

15. Van Buren, K.L., Mollineaux, M.G., Hemez, F.M., Atamturktur, S.: Simulating the dynamics of wind turbine blades: part II, model validation and uncertainty quantification. Wind Energy. **16**, 741–758 (2013)
16. Roache, P.J.: Perspective: a method for uniform reporting of grid refinement studies. ASME J. Fluids Eng. **116**, 405–413 (1994)
17. Stern, F., Wilson, R., Shao, J.: Quantitative V&V of computational fluid dynamics (CFD) simulations and certification of CFD codes with examples. In: 2004 ICHMT International Symposium on Advances in Computational Heat Transfer, Norway (2004)