Proceedings of the ASME 2010 3rd Joint US-European Fluids Engineering Summer Meeting and 8th International Conference on Nanochannels, Microchannels, and Minichannels
FEDSM-ICNMM2010
August 1-5, 2010, Montreal, Canada

FEDSM-ICNMM2010-31225

EFFECTIVE USE OF V&V20 SOLUTION VERIFICATION METHODOLOGY

Dr. Christopher J. Freitas
Computational Mechanics, Southwest Research Institute
San Antonio, Texas, USA

ABSTRACT

Methods for the quantification of numerical uncertainty have been a subject of interest to the American Society of Mechanical Engineers (ASME) and the mechanical engineering community as a whole for over a decade. During this time period, ASME has promulgated three statements of standards for the reporting of numerical uncertainty in archival publications (Journal of Fluids Engineering). This paper summarizes the work that has gone into the specification of these standards and the continuing effort in formulation of methods and procedures for quantifying numerical uncertainty. Specifically, this paper discusses the efforts of the ASME V&V 20 Committee (Verification and Validation in Computational Fluid Dynamics and Heat Transfer) to lay a foundation and structure to verification and validation for fluid flow and heat transfer simulations. Issues and methods related to code verification and in particular solution verification are presented and discussed in the context of the recently released V&V20 Standard.

INTRODUCTION

Verification is a process to establish and confirm code and solution accuracy. Validation is a process to establish numerical model accuracy in reference to physical data only. In the context of numerical simulation, we wish to ultimately establish and confirm the accuracy of a numerical model of a physical system. A numerical model consists of the code and the solution to a specific problem. The verification process for a numerical model must then establish and confirm accuracy for both the code and the solution. Code verification is distinct from Solution verification and must precede it, even though both procedures utilize grid convergence studies. In general, code verification assesses code correctness and specifically involves *error evaluation* for a known solution. By contrast, solution verification involves *error estimation*, since we generally do not know the exact solution to the specific problem. Code and solution verification are mathematical activities, with no concern whatever for the agreement of the numerical model with physical data from experiments, that is the concern of validation. Note, however, that the solution and its error estimate from a solution verification exercise will be used in the validation process. In this way, code verification, solution verification, and validation are coupled together into an overall process.

To support the formalization of an approach to the overall process of code verification, solution verification, parameter uncertainty assessment and ultimately validation of simulations, ASME established the Performance Test Code Committee, PTC-61: Verification and Validation in Computational Fluid Dynamics and Heat Transfer. PTC-61 held its first meeting in May 2004. (Note: ASME has subsequently changed the naming convention for Verification and Validation Committees under Codes and Standards to V&V##; PTC-61 is now V&V20) The members of this committee are:

Prof. Hugh W. Coleman (University of Alabama Huntsville) - Chairman
Dr. Christopher J. Freitas (Southwest Research Institute) – Vice-Chairman
Dr. Ben F. Blackwell (Consultant)
Dr. Kevin J. Dowding (Sandia National Laboratories)
Prof. Urmila Ghia (University of Cincinnati)
Prof. Richard G. Hills (New Mexico State University)
Dr. Roger W. Logan (Lawrence Livermore National Laboratory)
Dr. Patrick J. Roache (Consultant)
Prof. W. Glenn Steele (Mississippi State University)
Ryan Crane (ASME)

The charter of this committee is to "Provide procedures for quantifying the accuracy of modelling and simulation in computational fluid dynamics and heat transfer." An approach to verification and validation using experimental uncertainty analysis concepts to quantify the result of a validation effort was a key focus of the committee. With this focus, the V&V20 Standard was officially released in December 2009.

Presented in the remainder of this paper is a discussion of the key elements involved with code and solution verification and aspects of validation as presently conceived by the

Copyright © 2010 by ASME

Committee. **This paper is a summary of all the hard work performed by the entire V&V20 Committee.**

The purpose of this paper is to present some of the key details of primarily Solution Verification, but also the overall verification process as defined in the V&V20 Standard. Much of the information presented in this paper may be found in the V&V20 Standard, although new examples of the procedures are presented here. The ultimate objective of this paper is to help promulgate the methodology of V&V20 to the engineering community.

A CONCEPTUAL FRAMEWORK FOR VERIFICATION AND VALIDATION

Figure 1 provides a conceptual framework for code and solution verification, uncertainty assessment, and validation. In this conceptual framework, error may be viewed as hierarchical, in the sense that error may be associated with different levels of model formulation and implementation. In any assessment of error or uncertainty, it is implicit that we have a measure of reality (or nature) or at least a descriptive framework of reality. This is generally our starting or reference point. Certainly any measurement of reality also has uncertainty associated with it due to measurement inaccuracies, which is the justification and basis for experimental uncertainty methods. And further, the best that one can hope for in a validation exercise is for the regions bounded by uncertainties (measurement and computational) to be coincident to some measure.

With the assumption that we have a measure or description of reality, one may then formulate a Postulated Math Model (PMM). The PMM is a continuum model and is the model that we ultimately wish to demonstrate as providing some measured predictability of reality. The PMM may be described as the weak-sense model – the model form, such as the Reynolds-Averaged Navier Stokes equations. The PMM is then cast into a discretized form or the Computational Model (CM) – a model form sufficient for numerical simulation. The CM is a form of the strong-sense model which includes all the input parameters, boundary and initial conditions required to define a particular problem for simulation. The distinction between the weak-sense model and strong-sense model suggests that the validation at an experimental set-point occurs only for the strong-sense model, and that only through an ensemble of validated strong-sense models is the weak-sense model ultimately validated. Note that the PMM may also be defined in a strong-sense model form, but we choose to not do so in this paper. The key point in this discussion is that a weak-sense model may be validated by an ensemble of specific or strong-sense model validations.

As shown in Figure 1, the total error may be defined by the difference between reality (plus uncertainty) and the PMM, plus the difference between the PMM and a fully resolved CM. However, a fully resolved CM is not generally achieved, so the third term in the error equation results from an under-resolved CM. Each of these error terms contributes to the total error. Through the procedures of verification and validation supported by a method for quantifying uncertainty or the error band, one may then determine the magnitude of contribution by each term to the total error band. Based on the magnitudes of each term, an appropriate action would result. For example, if the contribution of the second term dominated the error, then we would know that an error was present in the code (programming or algorithmic). If the third term dominated the error, then this would suggest that the discretization scheme was not sufficiently accurate or some resolution dependent boundary condition was inappropriate, or that grid resolution in general was not sufficient. Through this systematic process, verification (code and solution) followed by validation may be achieved for a code and ultimately the PMM. The essential element to this process, however, is a method for computing uncertainty in the simulation. That subject is addressed later in this paper.

V&V20 has formalized the above into the following set of equations. We first define the validation comparison error $\boldsymbol{E}$ (also represented in Figure 1) as

$$\boldsymbol{E = S - D} \qquad \textbf{(1)}$$

Where $\boldsymbol{S}$ is the simulation solution and $\boldsymbol{D}$ is the experimental data. The error in $\boldsymbol{S}$ due to the accumulated errors as suggested in Figure 1, is defined as the difference between $\boldsymbol{S}$ and the true value $\boldsymbol{T}$, and similarly for the error in the experimental data $\boldsymbol{D}$, or

$$\boldsymbol{\delta_S = S - T} \quad \text{and} \quad \boldsymbol{\delta_D = D - T} \qquad \textbf{(2)}$$

The validation comparison error $\boldsymbol{E}$ is thus the combination of all the errors in the simulation solution and the experimental data, and its sign and magnitude are known once the validation comparison is made.

$$\boldsymbol{E = S - D = (T + \delta_S) - (T + \delta_D) = \delta_S - \delta_D} \qquad \textbf{(3)}$$

As illustrated in Figure 1, all errors in $\boldsymbol{S}$ can be assigned to one of three categories:

1. error due to modelling assumptions and approximations (errors in PMM), defined as δ_{Model},
2. error due to the discretization of PMM and numerical solution of the computational model CM, defined as δ_{Num}, and/or
3. error due to errors in the simulation input parameters, defined as δ_{Input}

Thus

$$\boldsymbol{\delta_S = \delta_{Model} + \delta_{Num} + \delta_{Input}} \qquad \textbf{(4)}$$

$$\boldsymbol{\delta_{Model} = E - \delta_{Num} - \delta_{Input} + \delta_D} \qquad \textbf{(5)}$$

The objective of the verification process is to estimate the error due to $\boldsymbol{\delta_{Num}}$, and the objective of the validation process is to estimate $\boldsymbol{\delta_{Model}}$. Equation 4 indicates that $\boldsymbol{\delta_{Input}}$ may be explicitly accounted for and thus contribute directly to the validation process and assessment of $\boldsymbol{\delta_{Model}}$ or it may be

Copyright © 2010 by ASME

implicitly part of δ_{Model}, which leads again to the distinction between a weak-sense model and a strong-sense model validation. In general, as discussed above, the weak-sense model refers to the mathematical model form or PMM, while the strong-sense model refers to the computational model (CM) with all associated boundary and initial conditions, and other input data required to define a particular problem. For the strong-sense model, δ_{Input}, need not necessarily be explicitly determined, but rather may be implicitly defined as part of δ_{Model} for a strong-sense model formulation. For example, consider a particular problem with a specified inflow boundary profile which in combination with the CM forms a strong-sense model. Simulation results are then compared to experimental data (validation) and an estimate of δ_{Model} is determined, which includes both model form errors and model input errors. This δ_{Model} is then specific to this strong-sense model with its specified inputs. If we now change the model inputs, we have created a new strong-sense model which may then be validated against the same or similar experimental data and another estimate of δ_{Model} is determined. We now have potentially (likely) two different solutions and two different model error estimates, which is appropriate since we have two different strong-sense models. In this example, δ_{Model} then has error contributions from both modelling assumptions and approximations (model form) and model input errors. In practice, there are numerous gradations or combinations of model form and model inputs contributing to δ_{Model}. Thus, it is crucial in interpreting the results of a validation effort that those error sources that are included in δ_{Model} be defined and understood – the strong-sense model must be well defined.

Equation 5 may also be written as the following for a strong-sense model in which model input error are considered fully part of the model form error

$$\delta_{Model} = E - \delta_{Num} + \delta_D \qquad (6)$$

Standard uncertainties for the errors on the RHS of equations 5 or 6 may be defined as u_{Num}, u_{Input}, and u_D, which represent the estimate of the standard deviation of the parent distribution from which the specific error estimate is a single realization. The error terms on the RHS of equations 5 or 6 and with their associated standard uncertainties define a validation uncertainty u_{Val} as follows

$$u_{Val} = \sqrt{u_{Num}^2 + u_{Input}^2 + u_D^2} \qquad (7)$$

From this we may then define the interval $E \pm u_{Val}$ within which δ_{Model} will fall with some unspecified degree of confidence. The estimation of u_{Val} is the kernel element of the V&V20 methodology, and E and u_{Val} are the validation metrics. The other kernel element in the methodology is the estimation of u_{Num}. The above discussion summarizes to a degree the overall framework to progress from verification through validation for a simulation and model. At this point, we now focus on code and solution verification only and ultimately the assessment of u_{Num}. The reader is directed to the V&V20 Standard for further discussions on the methods for assessing u_{Input} and u_D and other topics important to the estimation of E and u_{Val}.

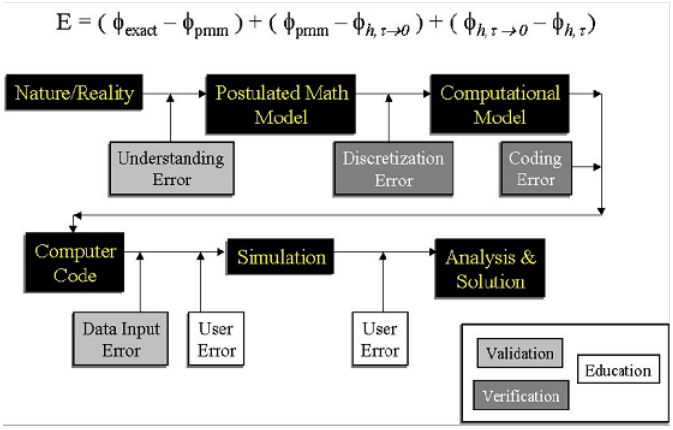


Figure 1. Conceptual framework for identification of sources of error in a computational modelling effort

CODE VERIFICATION

Code verification, establishing and confirming the correctness of the code itself can only be done by systematic discretization convergence tests and monitoring the convergence of the solutions towards a known benchmark solution. The best benchmark solution or standard of comparison is an exact analytical solution, ideally one expressed in simple primitive functions. Benchmark solutions involving infinite series are not desirable, due to potential difficulties in accurate evaluation of the benchmark solution itself. Further, it is not sufficient that the analytical solution be exact, i.e., the solution structure must also be sufficiently complex that all terms in the governing equation(s) and code being tested are exercised [1, 4]. A classic example of a physically meaningful analytical solution appropriate as a benchmark is the Sod problems – one-dimensional flow discontinuity problems or "shock tube" problems (although some may argue that these are not true analytical solutions since some iteration is required in the "analytic" solution process).

The process of developing a computer code (and the algorithms used in it) for nonlinear partial differential equations (PDE's) necessarily involves much testing and evaluation of algorithms and coding. Mostly, this is performed for sets of simplified problems with analytical solutions. For example, a 3D time-dependent full nonlinear Navier-Stokes code will probably have been tested on a simple 1-D linear advection-diffusion equation, a 2D or 3D Burgers equation, and other increasingly complex equation forms. While these tests are helpful in ascertaining code performance, and taken all together can constitute a partial or informal code verification, they are inadequate to convincingly demonstrate that the code is correct

Copyright © 2010 by ASME

for the targeted problems. In order to achieve convincing code verification, one needs an exact analytical solution or family of solutions that exercises all the important features of a code (e.g. variable properties, nonlinearities, turbulence models, etc.). It is well known that even the laminar Navier-Stokes equations do not have known analytical solutions for any but the most trivial boundary and initial conditions; in fact, the Navier-Stokes equations are recognized as one of the great unsolved problems of modern mathematics or physics. Fortunately, a very general procedure does exist for generating exact analytical solutions required for accuracy verification of codes. This procedure, the Method of Manufactured Solutions (MMS) [1, 2, 4] is presented in the next section.

Systematic grid or discretization convergence testing is based on a series of solutions generated by the computer code as a function of different grid resolutions, where the monitoring of convergence as $h \rightarrow 0$ is performed, where h is a measure of the level of discretization. Observed order of convergence is then based on the behaviour of the error of the discrete solution as $h \rightarrow 0$. In the most general sense, error is simply the difference between the discrete solution $f(h)$ and the exact solution F, $E = f(h) - F$. For an order P method, the error in the solution E asymptotically is proportional to h^P, or $E = f(h) - F = C\ h^P + H.O.T.$. A comparison between the values of the observed P and the theoretical P (from a Taylor series expansion for example) provides valuable insights to the numerical error in the computer code. If the values of the observed P and the theoretical P vary greatly from each other, then this indicates one of several possible issues:

- the grid convergence study has not been carried out to a sufficient level of refinement,
- there are more significant errors being generated in the code than those due to discretization and thus a detailed review of the code is required,
- boundary conditions and/or initial conditions may not be appropriate, and
- iterative convergence may not have been sufficient.

When a systematic grid convergence test is verified (for all point-by-point values), then we have verified

- equation transformations (e.g., nonorthogonal boundary fitted coordinates),
- order of the discretization,
- encoding of the discretization, and
- matrix solution procedures.

Figure 2 displays an illustration of what one would anticipate the computational solution should do as the grid is refined in a grid convergence study of an analytical solution, that is it approaches the analytic solution. In this example, the Sod problem is solved using a three-dimensional tetrahedral grid flow solver, in which a discontinuity in pressure and density of 100:1 is simulated. The characteristic length scale of the tetrahedral grids is systematically refined over 5 grid resolutions to a final length scale refinement of a factor of 3 in reference to the coarsest grid (G1 is the coarsest grid and G5 is the most refined grid in 3D). The axial distribution of pressure and sound speed are shown in Figure 2 as a function of grid resolution.

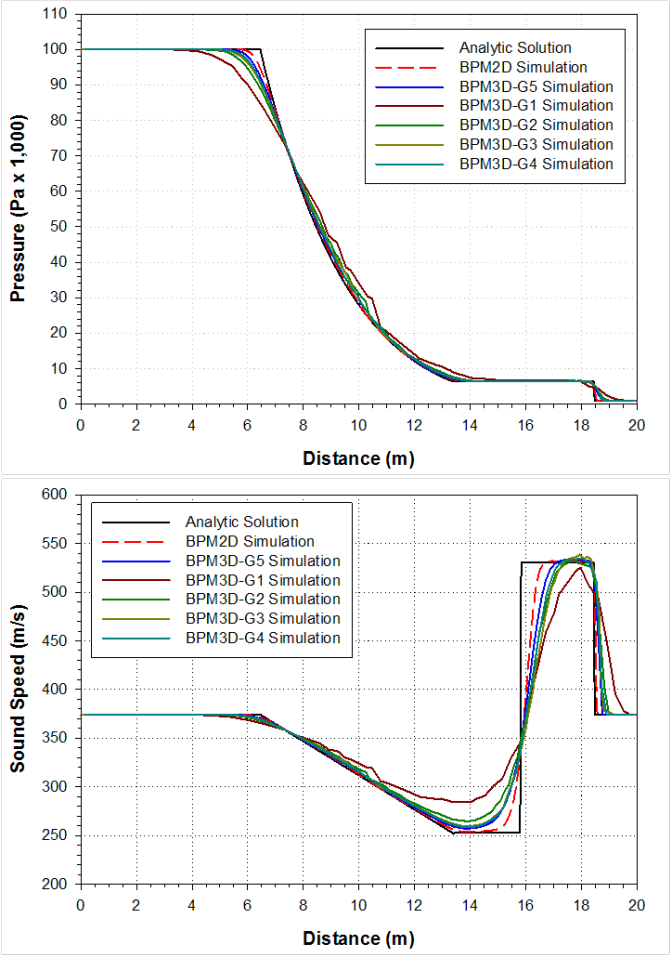


Figure 2. Example of grid convergence study, results compared to an analytical solution

METHOD OF MANUFACTURED SOLUTIONS

The Method of Manufactured Solutions (MMS) provides a methodology for code verification that has been successfully demonstrated in a variety of codes. It is only applicable, however, to codes based on the solution of partial differential equations. For some models, the method can be set up with no special code requirements, however, in general and the easiest way to apply MMS does require two code features that may not be already built in to the computer code (i.e., the ability to (1) introduce an arbitrary source term in to the code as well as (2) associated boundary conditions). The following brief discussion of MMS is given to provide a general sense of the method; detailed examples of the implementation of the method are given in the V&V20 Standard and references [1, 4].

Copyright © 2010 by ASME

Code Verification requires an *exact*, *analytical* solution to a non-trivial problem. The formulation of an exact, analytical solution may seem difficult for nonlinear systems of PDE's, but in fact it is relatively easy. MMS starts at the end, with a sufficiently complex solution form, e.g., hyperbolic tangents, which are easily evaluated and differentiated and contain all orders of derivatives (there are plenty of other functional forms with this attribute). Boundary conditions or domain structure are not necessarily important, just the solution form. Nor is "realism" a concern. Physical realism is not important to the "engineering proof" of code correctness, since only mathematics is involved. MMS simply must exercise all the terms in the mathematical model. In fact, *un*realistic solutions typically are better in this regard, since realistic solutions will often contain small parameters and boundary layers in which some terms are negligible, so that errors in these terms might go undetected.

In the generalized MMS approach, the problem is written symbolically as a nonlinear (system) operator L, L[f(x,y,z,t)] = 0. The manufactured solution M is then defined by f = M(x,y,z,t). We will now change the problem to a new operator L' such that the solution to L'[f(x,y,z,t)] = 0 is exactly the manufactured solution M. The most general and straightforward approach is to determine L' by adding a source term Q to the original problem, L' = L – Q, where the required source term is evaluated by allowing L to operate on the manufactured solution M, Q = L[M]. So instead of solving the original problem L(f) = 0 with an unknown solution, one solves the problem L(f) = Q (or equivalently, L'(f) = 0) which has the known solution M. Boundary values, for any boundary condition to be tested, are determined from M, as are the initial conditions. A detailed example of MMS application is provided in Appendix A in the V&V20 Standard.

Given this non-trivial exact analytical solution, we then perform grid convergence tests on the code and verify not only that it converges, but also at what rate it converges, the value of $\boldsymbol{P}$. Further, the magnitude (and sign) of the error is directly computed from the numerical solution and its comparison to the analytical solution. Again, in order to apply MMS in this most general form, it is necessary that the code be capable of handling source terms and non-homogeneous boundary conditions [3].

SOLUTION VERIFICATION

Systematic grid refinement is the cornerstone of verification processes for either codes or solutions. Whereas grid refinement studies in the context of code verification provide an *evaluation of error*, grid refinement studies used in solution verification provide an *estimate of error*. This estimate of error is determined through the prediction of uncertainty or uncertainty estimates, which are intended to bound the solution error. Two methods of obtaining solution uncertainty estimates from grid refinement studies are commonly used: classical Richardson Extrapolation [9] and Roache's Grid Convergence Index (GCI) [11]. GCI is in fact obtained from the (generalized) Richardson Extrapolation (RE) by multiplying the RE error estimate by an empirically determined Factor of Safety, $\boldsymbol{F_s}$. The $\boldsymbol{F_s}$ is intended to convert the 50% uncertainty (error band) implicit in the definition of any ordered error estimate (like RE) into a 95% uncertainty estimate.

Richardson Extrapolation is based on the assumption that discrete solutions f have a series representation in the grid spacing h. If the formal order of accuracy of an algorithm is known, then the method provides an estimate of the error when using solutions from two different (halved or doubled) grids. If the formal order of accuracy is not known, then three different (twice halved or doubled) grids and solutions are required to determine the order of the method and the error. Although grid doubling (or halving) is often used with Richardson Extrapolation, it is not required [1], and the ratio of grid spacing may be any real number.

Prior to the application of a grid convergence study for solution verification it is assumed that code verification has been completed and documented. In general, the computer code being used for the application being considered must be fully referenced, and previous verification studies must be described. Code verification will insure that the computer code is capable of solving a system of non-linear coupled partial differential equations with a properly posed set of initial and/or boundary conditions, and approaches the exact solution to these equations when a sufficiently fine grid resolution (both in time and space) is used.

Before any discretization error estimation is calculated, it must be ensured that iterative convergence (if iterative methods are used) is achieved. Iterative convergence of at least three orders of magnitude decrease in properly normalized residuals for each equation solved over the entire computational domain is often used. However, there currently is no justification for this level of convergence, and is based solely on users experience (and code default values) and has not been formally demonstrated. The more appropriate level of iterative convergence is when the iterative error has been reduced to a level negligible in reference to the discretization error. Recent results suggest that iteration error needs to be 2 to 3 orders of magnitude smaller than discretization error to guarantee negligible influence [13].

As stated above, the preferred method for uncertainty estimation based on discretization error is Richardson Extrapolation. Since its first elegant application by its originator, L. F. Richardson, in 1910 and later in 1927 [9, 10], this method has been studied by many authors. Its intricacies, pitfalls and generalizations have been exhaustively investigated. A short list of references are given and are selected for their direct relevance to the subject include; Celik et al [6], Celik and Karatekin [7], Eca and Hoekstra [8], Roache [1, 11], and Stern et al. [12].

Application of RE and GCI may encounter some difficulties in practical problems. When applied, local values of predicted variables may not exhibit a smooth, monotonic dependence on grid resolution, and in a time dependent

Copyright © 2010 by ASME

calculation, this non-smooth response will also be a function of time and space. But, nonetheless it is currently the most robust method available for the prediction of numerical uncertainty.

A PROCEDURE FOR ESTIMATING U_{NUM}

A five-step procedure is defined in the V&V20 Standard and is presented below for the application of the Grid Convergence Index (GCI) method.

Step 1:

Define a representative cell, mesh or grid size h. For example, for three dimensional, structured, geometrically similar grids

$$h = \left[(\Delta x_{\max})(\Delta y_{\max})(\Delta z_{\max})\right]^{1/3} \quad \textbf{(8)}$$

For non-structured grids one can define

$$h = ((\sum_{i=1}^{N} \Delta V_i)/N)^{1/3} \quad \textbf{(9)}$$

where ΔV_i is the volume of the i^{th} cell, and N is the total number of cells used for the computations [1]. Equation 9 was used to calculate the characteristic grid length scale used in Figure 2.

Step 2:

Select three significantly different sets of grid resolutions and run simulations to determine the values of key variables important to the objective of the simulation study, for example a variable ϕ. There are some advantages to using integer grid refinement, but it is not necessary. It is desirable that the grid refinement factor, $r=h_{coarse}/h_{fine}$, should be greater than 1.3 for most practical problems. This value of 1.3 is again based on experience and not on some formal derivation. The grid refinement should, however, be made systematically, that is, the refinement itself should be structured even if the grid is unstructured. Geometrically similar cells are preferable. It is highly recommended not to use different grid refinement factors in different directions, e.g. $r_x = 1.3$ and $r_y = 1.6$, because erroneous observed P values are produced, as shown by Salas [14]. (The computational solutions still converge to the correct answers with $r_x \neq r_y$ but the observed rate of convergence $\boldsymbol{P}$ is affected.)

Step 3:

Let $h_1 < h_2 < h_3$ and $r_{21} = h_2/h_1$, $r_{32} = h_3/h_2$ and calculate the apparent (or observed) order, $\boldsymbol{P}$, of the method from

$$p = (1/\ln(r_{21}))[\ln|\varepsilon_{32}/\varepsilon_{21}| + q(p)] \quad \textbf{(10)}$$

$$q(p) = \ln\left(\frac{r_{21}^{p} - s}{r_{32}^{p} - s}\right) \quad \textbf{(11)}$$

$$s = 1 \cdot \text{sign}(\varepsilon_{32}/\varepsilon_{21}) \quad \textbf{(12)}$$

where $\varepsilon_{32} = \phi_3 - \phi_2$, $\varepsilon_{21} = \phi_2 - \phi_1$, and ϕ_k denoting the simulation value of the variable on the k^{th} grid. Note that $q(p) = 0$ for r = *constant.* This set of three equations can be solved using fixed point iteration with the initial guess equal to the first term, i.e., $q = 0$.

A minimum of four grids is required to demonstrate that the observed order $\boldsymbol{P}$ is constant for a simulation series. The "right" three-grid solution for the observed order $\boldsymbol{P}$ may be adequate if some of the values of the variable ϕ predicted on the three grids are in the asymptotic region for the simulation series. In fact, it may require more than three grids to convincingly demonstrate asymptotic response in difficult problems, possibly five or six grid resolutions in cases where the convergence is noisy. This is all dependent on the initial grid resolution used and where the predicted value of ϕ lays as a function of grid resolution. However, in order to provide a balance between providing both a tractable method and insuring a level of accuracy in the predicted observed order $\boldsymbol{P}$, we propose that at least a three-grid study be performed.

Step 4:

Calculate the extrapolated values from the equation

$$\phi_{ext}^{21} = (r_{21}^{p}\phi_1 - \phi_2)/(r_{21}^{p} - 1) \quad \textbf{(13)}$$

and similarly for ϕ_{ext}^{31}.

Step 5:

Calculate and report the following error estimates along with the apparent order of the method $\boldsymbol{P}$, where the approximate relative error may be cast as a dimensionless form or a dimensioned form

dimensionless form: $$e_a^{21} = \left|\frac{\phi_1 - \phi_2}{\phi_1}\right| \quad \textbf{(14a)}$$

or, dimensioned form: $$e_a^{21} = |\phi_1 - \phi_2| \quad \textbf{(14b)}$$

Estimated extrapolated relative error is:

$$e_{ext}^{21} = \left|\frac{\phi_{ext}^{21} - \phi_1}{\phi_{ext}^{21}}\right| \quad \textbf{(15)}$$

The fine grid convergence index is finally:

Copyright © 2010 by ASME

$$\mathrm{GCI}_{\mathrm{fine}}^{21} = \frac{F_s \bullet e_a^{21}}{r_{21}^p - 1} \qquad \textbf{(16)}$$

$\boldsymbol{F_s}$ is the Factor of Safety which originally was assigned a value of 3 by Roache [11]. Roache [1] has subsequently recommended a less conservative value for $\boldsymbol{F_s}$ of 1.25 when using three grid studies and the observed $\boldsymbol{P}$. He arrived at this value through empirical studies and this value roughly correlates with the definition of uncertainty U used by Coleman and Stern [5] and suggests that using a value of 1.25 results in a GCI with a 95% confidence interval. Based on current evidence, we recommend a value of 1.25 be used with at least three grid studies.

If the calculated order of the method $\boldsymbol{P}$ is less than 1.0, an uncertainty band may also be given by assuming $\boldsymbol{P}$ = 1.0. We do this not to ignore the observed $\boldsymbol{P}$, but simply to give two calculations, one with the observed $\boldsymbol{P}$ and one with $\boldsymbol{P}$ = 1.0, as an indicator of the sensitivity of the uncertainty band to the observed value of $\boldsymbol{P}$. However, the GCI computed with the observed $\boldsymbol{P}$ is the more conservative approach. It should also be noted, that if the observed value of $\boldsymbol{P}$ is significantly different from the expected order of the method (for example the order of the method is believed to be third-order for the primary variables but it observed to be less than 1), than one should delve into the root cause of this difference. It may suggest a possible error in the method or its implementation, or that the grid resolutions used in the GCI analysis were not predicting values of the variable in the asymptotic region.

The form of the GCI is based on theory, but the use of absolute values for estimated errors and the factor $\boldsymbol{F_s}$ are based on empiricism involving the examination of several hundred CFD case studies. The empirical tests involved the determination of conservatism in 95% of the cases, corresponding to GCI = U_{num} = 95%. No assumptions on the form of the error distributions were made nor were necessary for these empirical studies, since actual data was examined with a simple pass/fail criterion. Equation 7 was developed using 1σ and the corresponding uncertainty $\boldsymbol{u_{Num}}$. In order to convert this (partially) empirical GCI from U_{num} to the $\boldsymbol{u_{Num}}$ needed in equation 7, it is now necessary to make the assumption of a Gaussian distribution. Then GCI = U_{num} = 95% corresponds to using a 2σ uncertainty, and the required term for use in equation 7 is then (see the V&V20 Standard for an additional discussion on this point):

$$\boldsymbol{u_{Num} = U_{num}/2 = GCI/2} \qquad \textbf{(17)}$$

CONSIDERATIONS IN SOLUTION VERIFICATION

The simulation variable ϕ that is evaluated by the five-step procedure can be any result of the simulation: local values of the dependent variables like *u*, *v*, *p*; volume-weighted RMS values; or integrated functionals of the solution like lift coefficient or heat flux. The same principles of solution verification apply in all cases, but the following should be noted. First, integrated functionals typically are better behaved (more smooth) than local values and thus the observed $\boldsymbol{P}$ tends to be less noisy. Second, different simulation variables can converge at different rates. Third, the same techniques for solution verification can be applied to derivatives of integrated functionals with respect to input parameters. Finally, care must be taken in determining the appropriate grid resolution requirements for both the grid convergence exercise and the grid resolution required to minimally resolve the physics of the problem. For example, if the problem to be solved has a specific range of length scales that characterize the flow physics such as boundary layers or thermal gradients, then the grid resolution for the coarsest grid used in the grid convergence study must still adequately resolve these length scales. This is particularly important in the context of Large Eddy Simulation. In Large Eddy Simulation, the filter width is usually related to a measure of the grid resolution, and thus as the grid resolution is changed during the grid convergence study, the filter width also is changed. This means that the partitioning of energy between the resolved and unresolved scales is changing. Thus if the user is not careful and as the grid convergence study is executed, they may be solving a different problem for some of the coarse grid resolutions if the cutoff between resolved and unresolved scales changes significantly from grid to grid.

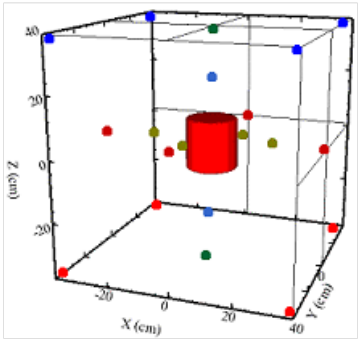
Finally, the following is suggested as an approach to effectively and efficiently perform and use a solution verification exercise in applications. For the given problem to be simulated, the first step is to define a set of simulation objectives (i.e., why are you simulating this problem, what quantities are you interested in predicting, and what level of accuracy is required). Given the simulation objectives, a nominal simulation problem is defined including boundary and initial conditions. This nominal problem should be representative of the problem set to be studied (where typically many simulations are performed to achieve the problem solution). This nominal problem will then serve as the basis for the solution verification grid convergence study. A detailed grid convergence study of this specific, nominal problem is executed with 3 to 6 levels of grid refinement. Based on the results of the solution verification for the nominal problem, a base grid resolution is defined that achieves the simulation objectives for estimated accuracy. This base grid resolution is then used in all subsequent simulations for the particular problem. If during the course of the subsequent simulations, the problem definition changes significantly such that the nominal problem no longer is representative of the study, then a new nominal problem should be defined and a new solution verification performed.

SOLUTION VERIFICATION EXAMPLES

Figure 3 provides an example of the five-step procedure defined above, where a TNT charge is detonated in a rigid,

Copyright © 2010 by ASME

fluid-filled box. The quantity of interest is the quasi-static pressure at various locations in the box (colored dots in top image) after a finite elapsed time in the time-dependent simulation. The bottom image in this figure displays the predicted value of pressure as a function of grid resolution at various measurement locations in the set of simulations. In this example, the magnitude of pressure has a smooth dependence on grid resolution. The basis for the grid resolution used is the number of zones across the diameter of the charge.



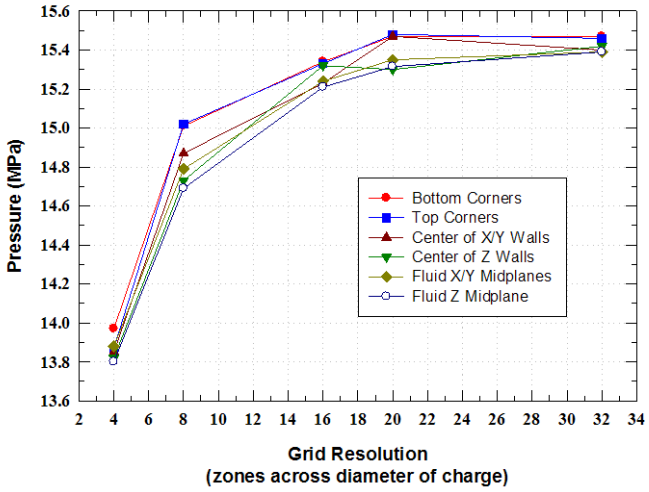


Figure 3. Sample Uncertainty Analysis

Table 1 summarizes the results of the application of the GCI to the explosive detonation problem. Here pressure at three different locations are used; i.e., a node in the corner of the box, a node near the center of a box side, and a node at mid-distance between the charge centerline and a box side. The second row of the table provides the computed (observed) order of the method, and the third row provides the computed error band or uncertainty. To compute these values the first four grid resolutions (4, 8, 16, and 20 zones across the diameter of the charge) were used. Rows four and five provide the range in pressure as predicted by the uncertainty estimate. This range should then bound the "exact" solution with a 95% confidence. The sixth row in the table displays the predicted value of pressure on the finest grid (resolution of 32 zones across the diameter of the charge). The ranges displayed in rows four and five should then bound the values here, and they do, again, demonstrating both the validity of this approach and the appropriateness of the magnitude of F_s. Please note that the units for pressure in Table 2 (rows 4 to 6) are MPa.

Table 1. Sample Uncertainty Analysis

Corner	Wall	Fluid
P = 1.7	P = 1.5	P = 1.02
GCI = 1.2%	GCI = 1.6%	GCI = 3.6%
15.34 ± 0.16	15.23 ± 0.21	15.24 ± 0.48
15.18 to 15.50	15.02 to 15.44	14.76 to 15.72
15.47	15.40	15.39

Another solution verification example is shown in Figure 4, where flow through an annular space bounded by an outer diameter tube and an inner hub is simulated (domain length of 1 cm). The inner hub has a wire wrapped around it, with diameter of D and represents the key geometric feature to be resolved in the flow simulation study.

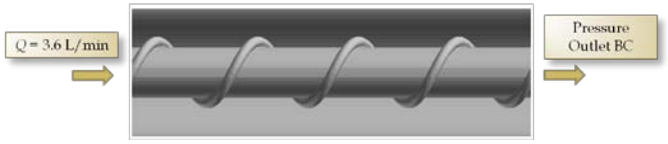


Figure 4. Annular flow geometry

Figure 5 shows the 5 grid resolutions used to perform the solution verification. Here the resolution of the wire wrap diameter ranges from 4 zones to 12 zones. The computer code used in this simulation uses a volume-of-fluid formulation which allows for the inclusion of solid objects embedded in the grid system without the need for body-fitted grid systems and the calculation of grid metrics. The resulting total number of grid cells used in these 5 resolutions are 0.4 M (0.24 M), 1.37 M (0.84 M), 3.07 M (1.91 M), 5.89 M (3.69 M), and 10.05 M (6.34 M), where the M stands for million, and the first number is the total number of grid cells in the domain, and the number in parenthesize is the fluid cells or cells open to fluid flow (volume fractions greater than 0).

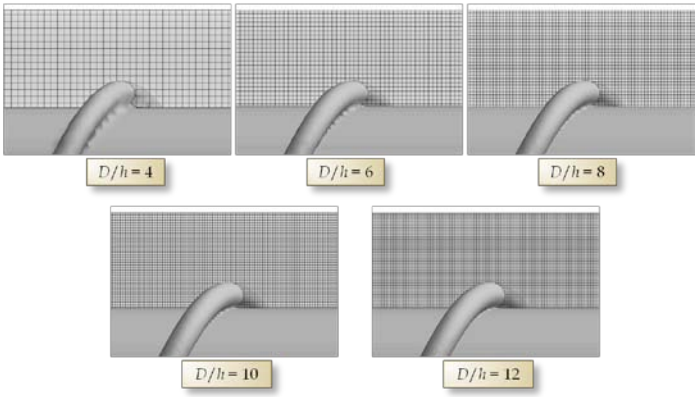


Figure 5. Grid resolutions used in solution verification (focused on a single wire wrap)

Copyright © 2010 by ASME

Figure 6 illustrates the affect of grid resolution on the distribution of fluid strain in the flow domain. Here contours of fluid strain on the surface of the hub and on the vertical centerline of the tube are shown for the same instance in elapsed time. Visualization of the key flow parameters can be a very useful supplementary set of data for establishing appropriate nominal grid resolution requirements for subsequent simulations. In this figure, for example, grid resolutions of D/h of 8 or greater are resolving quite similar fine-scale flow structures. In this example a nominal grid resolution of D/h = 8 was selected as the best balance between simulation accuracy and computational requirements (runtimes were only 30% and 20% respectively of the two finer grids).

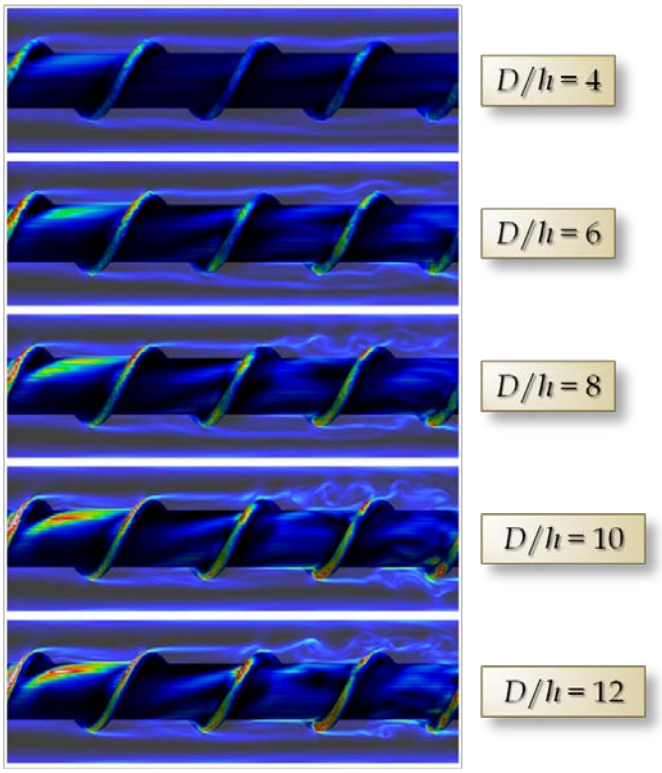


Figure 6. Fluid strain contours at different grid resolutions

For the conditions simulated in this problem, a nominal grid resolution of D/h = 8 was in the asymptotic region of the grid convergence curve when plotting the key parameter of total skin friction on the wire versus grid resolution. Total skin friction was the key parameter of interest in this particular analysis. At the grid resolution of D/h = 8, the GCI was 2%, which was more than sufficient for the application, and thus was the grid resolution selected for use in the subsequent production simulations and analyses.

ACKNOWLEDGMENTS

This paper has partially summarized the work performed and included in the V&V20 Standard. The author is a member (past Vice Chair) of the V&V20 committee. This paper would not have been possible without all the hard work of the entire V&V20 committee membership.

REFERENCES

[1] Roache, P. J., 1998, Verification and Validation in Computational Science and Engineering, Hermosa Publishers, Albuquerque, August 1998. http://www.hermosa-pub.com/hermosa.

[2] Oberkampf, W. L. and Trucano, T. G., 2002, "Verification and Validation in Computational Fluid Dynamics," Progress in Aerospace Sciences, vol. 38, no. 3, pp. 209-272.

[3] Roache, P. J., 2004, "Building PDE Codes to be Verifiable and Validatable," Computing in Science and Engineering, Special Issue on Verification and Validation, September/October, pp 30-38.

[4] Roache, P. J., 2002, "Code Verification by the Method of Manufactured Solutions", ASME Journal of Fluids Engineering, Vol. 114, No. 1, March, pp. 4-10.

[5] Coleman, H. W. and Stern F., 1997, "Uncertainties in CFD Code Validation," ASME Journal of Fluids Engineering, Vol. 119, pp. 795-803.

[6] Celik, I., Chen, C. J., Roache, P. J., and Scheuerer, G, eds., 1993, Proceedings: Symposium on Quantification of Uncertainty in Computational Fluid Dynamics, ASME Fluids Engineering Division Summer Conference, June 20-24, Washington, D.C., Vol. 158.

[7] Celik, I., and Karatekin, O., 1997, "Numerical Experiments on Application of Richardson Extrapolation With Nonuniform Grids," ASME Journal of Fluids Engineering, Vol. 119, pp.584-590.

[8] Eca, L. and Hoekstra, M., 2002, "An Evaluation of Verification Procedures for CFD Applications," 24th Symposium on Naval Hydrodynamics, Fukuoka, Japan, 8-13 July.

[9] Richardson, L. F., 1910, "The Approximate Arithmetical Solution by Finite Differences of Physical Problems Involving Differential Equations, with an Application to the Stresses In a Masonary Dam," Transactions of the Royal Society of London, Ser. A, Vol. 210, pp. 307-357.

[10] Richardson, L. F. and Gaunt, J. A., 1927, "The Deferred Approach to the Limit," Transactions of the Royal Society of London, Ser. A, Vol. 226, pp. 299-361.

[11] Roache, P. J., 1993, "A Method for Uniform Reporting of Grid Refinement Studies," Proceedings of Quantification of Uncertainty in Computation Fluid Dynamics, Edited by Celik, et al., ASME Fluids Engineering Division Spring Meeting, Washington D.C., June 230-240, ASME Publ. No. FED-Vol. 158.

[12] Stern, F., Wilson, R. V., Coleman, H. W., and Paterson, E. G., 2001, "Comprehensive Approach to Verification and Validation of CFD Simulations - Part 1: Methodology and Procedures," ASME Journal of Fluids Engineering, Vol. 123, pp. 793-802.

Copyright © 2010 by ASME

[13] Eça, L. and M. Hoekstra, M., 2007, "Evaluation of Numerical Error Estimation Based on Grid Refinement Studies with the Method of Manufactured Solutions," Report D72-42, Instituto Superior Tecnico, Lisbon.

[14] Salas, M.D., 2006, "Some Observations on Grid Convergence", Computers and Fluids, Vol. 35, Issue 7, August, pp. 688-692.

[15] Freitas, C.J., 1993, "Editorial Policy Statement on the Control of Numerical Accuracy", Journal of Fluids Engineering, Vol. 115, September, pp. 339-340.

Copyright © 2010 by ASME