Weapons Verification

Trevor Tippetts, Devin Shunk, Miles Buechler, Jason Pepin
W-13, Advanced Engineering Analysis
Los Alamos National Laboratory

September 4, 2009

Chapter 1

Verification

1.1 Introduction

Discretization error is introduced into the computed solutions when finite element discretizations represent partial differential equations. This error is often a significant component of the uncertainty in simulation results. Therefore, discretization error quantification via calculation verification, also known as solution verification, is vitally important.

Calculation verification resembles code verification in many ways. Code verification ascertains correctness of simulation codes by comparing a known solution of the governing partial differential equations (PDEs) to numerical solutions of discretized approximations. The computed rate of convergence in code verification is then compared to a theoretical value, known for the solution algorithm, as evidence of code correctness. Code verification is useful for testing simulation codes, but it is obviously limited to relatively simple simulations with analytical solutions.

Calculation verification refers to convergence analyses in which an exact solution is not used. Calculation verification can be a powerful tool to confirm convergence and to estimate discretization error. It is therefore an essential step in establishing confidence in simulation results.

Many discretization error estimation methods use an ansatz, *i.e.*, an assumed form for the discretization error. The assumption is justified by Richardson extrapolation,[1, 2] based on a series expansion of the error in a mesh scale parameter, Δx. Only the first term of the series is believed to be significant, and the truncated series represents the error. Extrapolation-based error estimators depend on the assumption that all of the meshes used to solve the PDEs are in this asymptotic region.

Previous work by Roache[3] used a generalized Richardson extrapolation to define the Grid Convergence Index (GCI). When errors convergence monotonically, the GCI is proportional to the Richardson-extrapolated error of the fine grid solution. The GCI uses a factor of safety based on analyst experience. Unfortunately, there is no guarantee that a new analysis is similar enough to Roache's previous analyses to justify a common safety factor. The GCI is not applicable when the error does not converge monotonically.

Stern, Wilson, and Shao[4] also used Richardson extrapolation to quantify discretization error as one step in a procedure for estimating uncertainty and intervals of certification for Computational Fluid Dynamics (CFD) codes. For the case of monotonic convergence, they evaluate the difference between Richardson extrapolation errors using estimated and theoretical rates of convergence. If the absolute value of that difference is large enough, it is defined to be the numerical uncertainty of the corrected solution. For smaller differences, the uncertainty is redefined so that it converges to ten percent of the extrapolation error. When convergence is oscillatory, they estimate uncertainty as half the difference between the maximum and minimum solution values. This approach does allow analysis of both monotonic and oscillatory convergence. However, the different definitions of uncertainty in the two cases can complicate interpretation of results. Stern *et al.* refer to some of the possible convergence conditions with solutions on three grids. Section 1.2 describes some additional cases that they did not identify.

A number of other authors have examined oscillatory convergence of discretization error. Celik, Li, Hu, and Shaffer studied numerical methods defined to have known convergence behavior for specific problems by construction.[5] They showed examples of oscillatory convergence and pointed out some of the possible problems that may arise from estimating the corresponding ansatz parameters. Their method of fitting splines to functions of error versus cell size is shown to give more accurate error estimates than a single-term ansatz for some cases. The large number of grids required could make it computationally prohibitive for many practical problems. Brock used an ansatz capable of modeling oscillatory error by assuming convergence of the absolute value of the error rather than a signed error.[6] He then used the triangle inequality to estimate an upper bound on the numerical error and convergence rate.

Kamm, Rider, and Brock[7] and Hemez[8] addressed the problem of combined space and time convergence with an ansatz that accounted for both space and time discretization. Their approach was applied to code verification analyses, using exact solutions to evaluate L_1 norms of the error. They demonstrated the method with a variety of problems, including both smooth and discontinuous solution fields.

It is possible to use an error ansatz at various points in a grid to obtain local information. Smitherman, Kamm, and Brock averaged the finite volume cells on finer grids to a common coarse grid.[9] As in Brock's paper[6], their local ansatz assumed convergence in absolute value of the error, allowing both monotonic and oscillatory convergence to use the same form. In examples of code and calculation verification analysis, the method was able to incorporate successfully the convergence rate of the oscillatory regions into the global convergence rate. In both of these works, an analytical expression was given for the three ansatz parameters for monotonic convergence and constant refinement ratio. Smitherman *et al.* used a nonlinear Newton solver for non-monotonic convergence. Newton's method could be applied to other ansatzes with different numbers of parameters, but it would always require equal numbers of grids and parameters because it must solve the linear system defined by a Jacobian matrix. Neither approach, therefore, could take advantage of data from additional grids.

Tippetts, Timmes, Brock, and Kamm extended this approach with a an optimization solver to minimize the ansatz fit residuals.[10] This approach allows more than the minimum number of three meshes to fit a single ansatz. Three meshes produce the same results as Newton's method, as all residuals will go to zero. However, with four or more meshes, the non-zero residuals provide information on the quality of fit. This information indicates the validity of the asymptotic assumption.

Roy and Raju[11] have developed the Method of Nearby Problems (MNP) as an alternative to error estimators based on Richardson extrapolation. The method uses a curve fit of a numerical solution as the exact, analytic solution to a "nearby problem." The analytical PDEs are solved on the nearby problem to generate a source term, so that the exact solution and exact error of this new problem are known. MNP implementation requires the same capability for arbitrary source terms as with the Method of Manufactured Solutions (MMS).[12] Many simulation codes lack this feature, and it could be costly or impossible to implement. It is also not always clear how best to perform the curve fits for many problems. For example, models with shock load, contact and friction, and nonlinear materials would be difficult to solve, especially for fields in multiple dimensions. Error estimates generated with MNP are only truly applicable to the nearby problem, not the original problem. A modeling error of unknown magnitude remains, due to the difference in source terms.

A few authors have made preliminary attempts at code verification with the Abaqus and Paradyn finite element codes.[13, 14, 15] Much work remains to characterize convergence for all quantities with all simulation physics. This work focuses on Calculation Verification for systems of interest to LANL and W-13. It is hoped that the practical experience gained and importance demonstrated by these results will motivate future work on establishing ideal rates of convergence for these codes.

In recent years Hemez[16] has developed a new variation on Richardson extrapolation methods. Rather than analysing convergence on the data directly, Functional Convergence methods project simulation outputs onto orthogonal components. The components are derived with a Singular Value Decomposition (SVD) of a data matrix. Hemez proved that convergence of the component magnitudes is a necessary and sufficient condition for convergence of the L_2 norm of the error. Functional Convergence has demonstrated improved robustness when non-convergent sources contribute to the discretization error.

In this verification analysis, the error absolute value is assumed to converge exponentially in the mesh size, as in some previous works.[6, 9, 16, 10] Following Tippetts et al.[10] an optimization problem enables the solution of the ansatz parameters. Implicit and explicit analytical expressions for the ansatz parameters are given for the special case of solutions from three meshes, for both monotonic and oscillatory convergence.

1.2 Calculation Verification Approach

1.2.1 Error Ansatz Solution

A typical error ansatz is

$$|\hat{y} - y_i| = A \left(\frac{\Delta x_i}{\Delta x_r} \right)^q . \tag{1.1}$$

In Equation 1.1, $\hat{y}$ is the extrapolated field value, and y_i is the field value given by the simulation code on mesh i, which has the characteristic element size Δx_i. The element size Δx_i is normalized by a reference mesh size Δx_r. This normalization makes the prefactor coefficient, A, equal to the error on mesh r and gives it the same units as y, regardless of q. The rate of convergence is determined by the exponent q, and the $|\cdot|$ functional operator is an L_p norm with an integral approximation and domain that are defined by the ansatz implementation.

A calculation verification analysis using Equation 1.1 would therefore have three unknown parameters for which to solve ($\hat{y}$, A, and q), requiring y_i and Δx_i from at least three meshes. The parameters may be defined as the solution to an optimization problem, with a cost function

$$f^2 = \sum_{i=1}^{N} \left(|\hat{y} - y_i| - A \left(\frac{\Delta x_i}{\Delta x_r} \right)^q \right)^2 , \tag{1.2}$$

which is minimized over the domains of $\hat{y}$, A, and q. If there are three meshes ($N = 3$), then the fit error, f, should be zero. The fit error will be greater than zero when N is greater than three.

Parameter Initialization

The expression to minimize in Equation 1.2 has discontinuous derivatives and, in many cases, many local minima. These properties can cause difficulty for optimization solvers. The final solution for the ansatz parameters can be very sensitive to the initial values used in the minimization search. The optimization algorithm might not converge to any solution if the initialization is too far from a minimum. This section describes a semi-analytical solution to Equation 1.1 for the case of three meshes. The analytical solution may be applied to a set of three fine meshes to provide the optimization solver with good initial values.

For the L_1 norm, the ansatz models the integral of the absolute value of the error. Because the inverse of the absolute value function is multivalued, a set of three pairs for y_i and Δx_i may have multiple solutions to Equation 1.1. The solutions may be distinguished from each other by accounting for the sign of the absolute value argument,

$$\hat{y} - y_i = s_i A \left(\frac{\Delta x_i}{\Delta x_r} \right)^q . \tag{1.3}$$

In Equation 1.3, s_i is the sign of the error on mesh i and is either 1 or -1. Defining R as the following ratio of solutions on coarse (c), medium (m), and fine (f) meshes,

$$R = \frac{y_f - y_m}{y_m - y_c}, \tag{1.4}$$

this representation leads to an implicit solution[1] for q that satisfies the equation

$$R = \frac{1 - s_f s_m \left(\frac{\Delta x_f}{\Delta x_m} \right)^q}{s_m s_c \left(\frac{\Delta x_f}{\Delta x_m} \right)^{-q} - 1}. \tag{1.5}$$

If $\Delta x_f / \Delta x_m = \Delta x_m / \Delta x_c$, Equation 1.5 becomes quadratic in $(\Delta x_m / \Delta x_c)^q$ and may be solved explicitly for q. Otherwise, a one-dimensional optimization produces the correct value.

Once q is known, A and y easily follow:

$$A = \Delta x_r^q \left| \frac{y_i - y_j}{s_j \Delta x_j^q - s_i \Delta x_i^q} \right| \tag{1.6}$$

and

$$\hat{y} = y_i + s_i A \left(\frac{\Delta x_i}{\Delta x_r} \right)^q . \tag{1.7}$$

When only three meshes are used in the verification analysis, Equations 1.5, 1.6, and 1.7 give the ansatz parameters directly. There is no need for any further numerical optimization or nonlinear system solver. If the ansatz is to be fit to four or more solutions, Equations 1.5 through 1.7 may be used to generate initial values for an optimization solver. Each combination of three solutions will produce a unique set of initial values for the optimization solver for f. All possible combinations were used to initialize the solver for the results in this work. The reported result is the one that ended with the smallest f, with preference given to convergent fits.

[1]Note that Equation 5 in [10] contained a similar expression for q with a sign error. The error has been corrected in Equation 1.5

Table 1.1: Critical values for R given q, $s_f s_m$, and $s_m s_c$.

$\lfloor s_f s_m, s_m s_c \rfloor$	$q = -\infty$	$q = 0^-$	$q = 0^+$	$q = \infty$
$\lfloor\ 1,\ \ 1\rfloor$	∞	$\frac{\ln \Delta x_f - \ln \Delta x_m}{\ln \Delta x_m - \ln \Delta x_c}$	$\frac{\ln \Delta x_f - \ln \Delta x_m}{\ln \Delta x_m - \ln \Delta x_c}$	0
$\lfloor -1,\ \ 1\rfloor$	$-\infty$	$-\infty$	∞	0
$\lfloor -1, -1\rfloor$	$-\infty$	-1^-	-1^+	0
$\lfloor\ 1, -1\rfloor$	∞	0	0	0

As noted above, there are multiple solutions to Equation 1.5 because each of s_f/s_m, s_c/s_m, and s_q may be equal to 1 or -1. However, some solutions give complex values for q. Other solutions have q equal to zero and $s_c = s_m = s_f$, which leads to infinite A and y. Still others give negative values for q, indicating divergence. The number of allowable solutions is reduced by applying the constraint that q must be real, and it may be reduced further by preferring convergent to nonconvergent solutions. The imaginary part and sign of q depend on the value of R. Table 1.1 lists the values of R corresponding to the divergent and convergent intervals of q.

Table 1.1 shows that for some values of R, there are multiple convergent solution On other R intervals, there is only one, and there are none for $R < -1$. The $s_f s_m = s_m s_c = -1$ solution for $R < 0$ has the advantage of continuity in parameters with respect to R, through both the convergent interval $-1 < R < 0$ and the divergent interval $R < -1$. For $0 < R < \frac{\ln \Delta x_f - \ln \Delta x_m}{\ln \Delta x_m - \ln \Delta x_c}$, $s_f s_m = s_m s_c = 1$ is preferred in this work because it is the only solution in Table 1.1 that reflects monotonic convergence.

The choice of solutions to use in these cases may be somewhat arbitrary. An alternate approach would be to include all possible convergent solutions in an expression for error range. Physical constraints on the extrapolated value might also eliminate some of the solutions. However, for simplicity this work will use a single solution for each ansatz according to the reasoning stated above.

1.2.2 Functional Convergence

The main idea behind the Functional Convergence method is that the data are well represented by components that are uncorrelated over the domain of interest. When the data are assembled in a matrix, the left singular vectors from the SVD are the orthogonal components that minimize successive reprojection error.

$$Y = \begin{bmatrix} y_1(\Delta x_c) & y_1(\Delta x_m) & y_1(\Delta x_f) \\ \vdots & \vdots & \vdots \\ y_N(\Delta x_c) & y_N(\Delta x_m) & y_N(\Delta x_f) \end{bmatrix} = U\sigma V^T \tag{1.8}$$

Ansatz fits and error estimation are applied to the magnitudes of the components for each of the meshes, defined by Hemez[16] as "generalized coordinates" (GCs), η.

$$\eta = V\sigma \tag{1.9}$$

If the GCs with greater magnitude are convergent, smaller non-convergent errors in in smaller GCs do not interfere with the ansatz fit. This allows for a potentially greater robustness when the ansatz parameters are very sensitive to non-convergent perturbations added to an otherwise convergent error. Orthogonality allows a simple expression of the error L_2 norm in terms of the GCs.

$$\begin{aligned} \|\hat{y} - y(\Delta x)\|^2 &= (\hat{y} - y(\Delta x))^T (\hat{y} - y(\Delta x)) \\ &= (\hat{\eta} - \eta(\Delta x))^T U^T U (\hat{\eta} - \eta(\Delta x)) \\ &= \textstyle\sum_k (\hat{\eta}_k - \eta_k(\Delta x))^2 \end{aligned} \tag{1.10}$$

It is clear from Equation 1.10 that convergence in the error L_2 norm is equivalent to convergence of all of the GCs.

Time Histories

Equation 1.8 weights each of the discrete data y_i equally. If the optimal orthogonal vectors U are believed to exist in a physical domain, it can only apply to regular samples. For example, a time history would have to be sampled on regular intervals and a field in space must be from a perfectly structured mesh. A slight modification to Equation 1.8 is necessary to generalize the approach for irregular samples and unstructured meshes.

A data matrix D composed of inner products of pairs of data vectors, $D_{ij} = y(\Delta x_i)^T y(\Delta x_j)$, has eigenvalues and eigenvectors equal to the squared singular values and right singular vectors of Y.

$$D = Y^T Y = V \sigma^2 V^T = \eta \eta^T \tag{1.11}$$

An analogous data matrix emerges from an L_2 norm and inner product suitably defined on a continuous time domain.

$$\|f\|^2 = \langle f, f \rangle \tag{1.12}$$

$$\langle f, g \rangle = \int f(t) g(t) dt \tag{1.13}$$

Irregularly sampled time histories are interpolated for the integral in Equation 1.13. The results to follow used a simple linear interpolation, which should be sufficient as long as the frequency content is small near the Nyquist limit. The ansatz fits and error estimation proceeds as usual on the GCs obtained from the eigen decomposition of D. Rather than left singular vectors, the orthogonal components of $y(t; \Delta x_i)$ are continuous functions of time, $U_k(t)$.

Field Variables

Functional Convergence for fields in a spatial domain is closely analogous to the development for time histories. Now the inner product is defined by the model geometry.

$$\langle f, g \rangle = \int f(x) g(x) dV \approx \sum_n w_n f(x_n) g(x_n) \tag{1.14}$$

For finite element models, the integrand in Equation 1.14 is a piecewise polynomial on a geometrically complex domain with arbitrary polyhedra defined by the faces of all elements on all meshes. A volume-weighted sum of field sample points therefore approximates the integrals. Two strategies choose the sample points x_n. First, the union set of all nodes from all meshes are included. Second, a random location in the interior of each element from all meshes is chosen. The second pass improves the integral estimate and can be repeated as a Monte Carlo procedure to converge to the exact integral. The weights, w_n, are derived from the fraction of total volume enclosed by the element or elements associated with point x_n. The orthogonal fields $U_k(x)$ are now functions of the spatial variables.

It is necessary for this procedure to interpolate the field on each mesh to a common set of points. The mapper[17] program, written by R. Robert Stevens, was an enabling code to meet this need for this project.

1.2.3 Metrics

The two key reasons for performing calculation verification are to demonstrate convergence of the simulation response quantities and estimate their uncertainty. With Richardson extrapolation methods, these two activities are closely tied. The three ansatz parameters $\hat{y}$, A, and q provide a means for estimating the error on any of the meshes. These estimates are likely useful only if the rate of convergence is reasonable. For example, if the fit gave an exponent of 15, the corresponding error estimates would not be meaningful unless the code developers really did implement an algorithm with 15^{th} order accuracy. The same is true if the exponent is negative, since that would imply greater error with increased refinement. (Note that this is not the same as estimating quantities that are expected to diverge. A different ansatz should be used in these cases since $\hat{y}$ presumes a finite extrapolation.) Most algorithms in typical finite element applications give first or second order convergence. It is unreasonable to expect the estimated exponent to match closely the theoretical value for most quantities on practical simulations. Unlike code verification models, these models are not specifically designed to isolate sources of error, and it is the author's anecdotal experience that the exponent is particularly sensitive to small changes in the simulation outputs. For this work, exponents in the interval $[0.5, 3]$ are considered convergent.

If all GCs are convergent, the sum of the components U_k scaled by extrapolated GCs $\hat{\eta}_k$ is equal to $\hat{y}$. However, this only gives a single estimate for the error, $\hat{y} - y$, which is not a very complete expression of uncertainty. It also fails to handle the very common case in which at least one component is not convergent. In many applications, a distribution of estimated $\hat{y}$ would be very useful for the analysts and stakeholders. Section 1.2.3 shows one example of how three additional assumptions yield a bootstrap resampling procedure to generate such a distribution.

Bootstrap and Residuals

There will be an in-sample residuals, r_i, for any ansatz fit with more than three meshes.

$$\hat{\eta} = \eta_i + s_i A \left(\frac{\Delta x_i}{\Delta x_r} \right)^q + r_i \tag{1.15}$$

It is reasonable to assume that whatever factors caused an imperfect ansatz fit might have been positive or negative and affected any of the simulations in the same way. This conjecture leads to a resampling scheme that randomizes the signs and magnitudes of residuals between meshes.

$$\hat{\eta}_{\pm ij} = \eta_i + s_i A \left(\frac{\Delta x_i}{\Delta x_r} \right)^q \pm r_j \tag{1.16}$$

Assumptions about distributions for $\hat{\eta}$ on a non-convergent modes are probably more tenuous, but they might be less consequential when the non-convergent errors are small in magnitude relative to the convergent errors. By definition, non-convergent components do not decrease in magnitude with mesh refinement. This suggests the simple assumption that a non-convergent GC, $\eta(\Delta x_i)$, is a random varible with independent identical distributions for each mesh. The cornerstone of the bootstrap method is the assumption that an observed distribution is a good approximation of the underlying distribution. Therefore, the $\eta(\Delta x_i)$ are used as a resample distribution for $\hat{\eta}$ on non-convergent components in the uncertainty estimator in this work.

The third key assumption in the bootstrap estimator is that the distributions of $\hat{\eta}_k$ for each component k are independent. Each distribution for $\hat{\eta}_k$ has the effect of "smearing out" the estimates of the others when they added back together. The resulting probability distribution for $\hat{y}$ could then answer real-world questions such as, "What are the 90% confidence intervals for this time history?" or "What is the probability of exceeding this failure criterion?"

With more than three meshes, it possible not only to use all of them at once for a single estimate of $\hat{y}$, A, q, and r_i. It is also possible to take combinations of fewer meshes for additional parameter estimates and out-of-sample residuals. Results from this mesh combination sampling will be shown, but the possible uncertainty information has not yet been incorporated into the resampling uncertainty estimator. There are many other possible improvements to the resampling technique described above, including refitting after resampling residuals and using symmetric distributions for $\eta(\Delta x_i)$. It should also be acknowledged that the statistics could be very poor with so few samples. The uncertainty estimator is presented only as one example of how the esoteric component ansatz parameters can translate back into the domain of interest.

Normalization and convergent fractions

The results in the following sections are normalized to facilitate their interpretation. In all cases, GC errors and ansatz parameters are normalized by the L_2 norm of the extrapolated solution, $\hat{y}$.

$$\|\hat{y}\| = \frac{A_j}{\sqrt{\Sigma_k \hat{\eta}_k^2}} \left(\frac{\Delta x_i}{\Delta x_r}\right)^q \tag{1.17}$$

$$\frac{|\hat{\eta}_j - \eta_j(\Delta x_i)|}{\sqrt{\Sigma_k \hat{\eta}_k^2}} = \frac{A_j}{\sqrt{\Sigma_k \hat{\eta}_k^2}} \left(\frac{\Delta x_i}{\Delta x_r}\right)^q + \frac{r_{jk}}{\sqrt{\Sigma_k \hat{\eta}_k^2}} \tag{1.18}$$

This normalization makes all errors and residuals into dimensionless fractions. The root-mean-square of the component errors will equal the relative error L_2 norm. This relative error can be large for quantities that should converge to zero, but the same expression is used for consistency.

Since not all components are convergent, another useful metric is the convergent error fraction, e_c, defined as the L_2 norm of the convergent error components divided by the L_2 norm of the total error.

$$e_c = \sqrt{\frac{\sum_{m \in convergent} \left(\hat{\eta_m} - \eta_m\right)^2}{\sum_k \left(\hat{\eta_k} - \eta_k\right)^2}} \tag{1.19}$$

An e_c of 1 is desirable, as it indicates that all component errors are well charaterized by the error ansatz. The utility of the convergent error fraction is most clear in the following hypothetical example: Suppose that a quantity with only one convergent component has a higher e_c than another quantity with only one non-convergent component. This is possible because e_c depends on the relative magnitudes of the component errors, not just on how many are convergent. Intuitively, it is the fraction of the error that is captured by the ansatz, and is really an evaluation of the ansatz

Table 1.2: Differences between Legacy and Baseline models.

	Legacy	Baseline
Time to create model	Long	Short
Easily adaptable?	No	Yes
Calculation verification?	No	Yes
Cross-code support?	No	Yes
Revision control?	No	Yes

itself. While it might often be the case that pathological quantities with no discernable convergence have both small e_c and large estimated error, a response quantity with a small e_c might have a small error, and vice versa.

1.3 Baseline model motivation

The Advanced Engineering Analysis group (W-13) at LANL provides the Nuclear Weapons Program with answers to the many complex engineering problems that arise during a weapon's life, including the stockpile-to-target (STS) sequence and various types of hostile and abnormal environments. In order to answer these problems, W-13 performs numerical simulations in diverse technical fields that include structural dynamics, radiation transport, thermodynamics, hydrodynamics, and probabilistic mechanics. These simulations require the construction and use of large-scale finite element models. To become more responsive in answering weapon system questions, W-13 has recently begun building a baseline model capability that greatly enhances the analyst's ability to assemble and run simulations.

Over the last two years, W-13 has been transitioning from legacy finite element models to newer baseline models. Legacy models have historically been built by an analyst to answer one or two problems. These models are usually not shared between analysts and are very difficult to modify due to the meshing scheme used. Baseline models are fully parameterized, meaning the part geometries are scripted and finite element models are generated based on additional scripts. Baseline models are not necessarily higher fidelity, i.e. "better", than the legacy models, but they are more easily adapted to various problems of interest and different finite element codes. Table 1.2 summarizes some of the major differences between baseline and legacy weapon models.

The modernization of W-13 weapon models from legacy to baseline status set the stage for the calculation verification milestone documented in this report. Parameterized scripts allow analysts to easily develop models of various mesh densities and execute calculation verification on large-scale models. To our knowledge, this is the first time that calculation verification has been attempted on finite element models of this size. In the past, the engineering V&V program within W-13 carried out validation experiments in order to quantify the accuracy of weapon system models. The recent development of baseline models allows us to complete the second piece of the puzzle, quantification of numerical error in these models through verification techniques.

Energy (J)Classified Content will go here.

Bibliography

[1] L. F. Richardson. The approximate arithmetical solution by finite differences of physical problems involving differential equations with an application to the stresses in a masonry dam. *Transactions of the Royal Society of London*, Ser. A(210):307 – 357, 1910.

[2] L. F. Richardson and J. A. Gaunt. The deferred approach to the limit. *Transactions of the Royal Society of London*, Ser. A(226):299 – 361, 1927.

[3] Patrick J. Roache. *Verification and Validation in Computational Science and Engineering.* Number 0913478083. Hermosa Publishers, 1998. QA297 .R63 1998.

[4] F. Stern, R. Wilson, and J. Shao. Quantitative V&V of CFD simulations and certification of CFD codes. *International Journal for Numerical Methods in Fluids*, 50(11):1335 – 55, 2006.

[5] I. Celik, L. Jun, G. Hu, and C. Shaffer. Limitations of Richardson extrapolation and some possible remedies. *Journal of Fluids Engineering*, 127:795 – 805, July 2005.

[6] J. S. Brock. Bounded numerical error estimates for oscillatory convergence of simulation data. In *18th AIAA Computational Fluid Dynamics Conference*, number AIAA-2007-4091, June 2007.

[7] J. R. Kamm, W. J. Rider, and J. S. Brock. Combined space and time convergence analyses of a compresible flow algorithm. In *16th AIAA Computational Fluid Dynamics Conference*, number AIAA-2003-4241, June 2003. LA-UR-03-2628.

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

[9] D. P. Smitherman, J. R. Kamm, and J. S. Brock. Calculation verification: pointwise estimation of solutions and their method-associated numerical error. *Journal of Aerospace Computing, Information, and Communication*, 4:676 – 692, March 2007. LA-UR-06-2006.

[10] T. Tippetts, F. X. Timmes, J. S. Brock, and J. R. Kamm. Field-wide Calculation Verification for Finite Volume Hydrodynamics Simulations. In *18th AIAA Computational Fluid Dynamics Conference*, number AIAA-2007-4090, June 2007. LA-UR-07-4009.

[11] C. J. Roy, A. Raju, and M. M. Hopkins. Estimation of discretization errors using the method of nearby problems. *AIAA Journal*, 45(6):1232 – 1243, June 2007.

[12] P. M. Knupp and K. Salari. *Verification of Computer Codes in Computational Science and Engineering.* Chapman and Hall, 2002.

[13] Miles Buechler, Amanda McCarty, Derek Reding, and Ryan Maupin. Explicit finite element code verification problems. Technical report, Los Alamos National Laboratory.

[14] T. Tippetts. Code-to-code comparison: Abaqus and adagio. Technical Report W-13-TR-0014U, Los Alamos National Laboratory, 2008.

[15] D. Castaño. Verification test suite for ASCI codes. Technical report, Los Alamos National Laboratory.

[16] F.M. Hemez. Functional data analysis of solution convergence. Technical Report LA-UR-07-5758, Los Alamos National Laboratory, 2007.

[17] R.R. Stevens. Mesh-to-mesh mapping of analysis results. Technical report, Los Alamos National Laboratory.