Extrapolation-based Discretization Error and Uncertainty Estimation in Computational Fluid Dynamics

Tyrone S. Phillips

Thesis submitted to the faculty of the Virginia Polytechnic Institute and State University in partial fulfillment of the requirements for the degree of

Master of Science
In
Aerospace Engineering

Christopher Roy, Chair
Danesh Tafti
William Mason
Eugene Cliff

February 27, 2012
Blacksburg, VA

KEYWORDS: Richardson Extrapolation, Grid Convergence Index, Euler, Navier-Stokes, Finite Volume Method

Extrapolation-based Discretization Error and Uncertainty Estimation in Computational Fluid Dynamics

Tyrone S. Phillips

ABSTRACT

The solution to partial differential equations generally requires approximations that result in numerical error in the final solution. Of the different types of numerical error in a solution, discretization error is the largest and most difficult error to estimate. In addition, the accuracy of the discretization error estimates relies on the solution (or multiple solutions used in the estimate) being in the asymptotic range. The asymptotic range is used to describe the convergence of a solution, where an asymptotic solution approaches the exact solution at a rate proportional to the change in mesh spacing to an exponent equal to the formal order of accuracy. A non-asymptotic solution can result in unpredictable convergence rates introducing uncertainty in discretization error estimates. To account for the additional uncertainty, various discretization uncertainty estimators have been developed.

The goal of this work is to evaluation discretization error and discretization uncertainty estimators based on Richardson extrapolation for computational fluid dynamics problems. In order to evaluate the estimators, the exact solution should be known. A select set of solutions to the 2D Euler equations with known exact solutions are used to evaluate the estimators. Since exact solutions are only available for trivial cases, two applications are also used to evaluate the estimators which are solutions to the Navier-Stokes equations: a laminar flat plate and a turbulent flat plate using the $k-\omega\ SST$ turbulence model. Since the exact solutions to the Navier-Stokes equations for these cases are unknown, numerical benchmarks are created which are solutions on significantly finer meshes than the solutions used to estimate the discretization error and uncertainty. Metrics are developed to evaluate the accuracy of the error and uncertainty estimates and to study the behavior of each estimator when the solutions are in, near, and far from the asymptotic range.

Based on the results, general recommendations are made for the implementation of the error and uncertainty estimators. In addition, a new uncertainty estimator is proposed with the goal of combining the favorable attributes of the discretization error and uncertainty estimators evaluated. The new estimator is evaluated using numerical solutions which were not used for development and shows improved accuracy over the evaluated estimators.

Acknowledgements

I sincerely thank Dr. Chris Roy for his guidance, encouragement, and patience in helping me develop research skills necessary to successfully complete my Master's degree. I am also very grateful for his continuous financial support.

I thank Dr. Eugene Cliff, Dr. Danesh Tafti, and Dr. William Mason for serving on my committee.

I am very grateful for my wife Larissa and for her unwavering love and support. I am also very thankful for my grandparents, parents, and brother for their continuous encouragement.

Table of Contents

List of Figures

List of Tables

1 Motivation

Fluid dynamics is important for the design of aerospace systems. Several tools and methods exist including analytic fluid dynamics , experimental fluid dynamics, and Computational Fluid Dynamics (CFD), all of which have advantages and disadvantages that the fluid dynamicist must consider.

Analytic fluid dynamics uses exact solutions or similar solutions derived from simplifying assumptions of specific flow equations. For example, a similar solution originally introduced by Blasius in 1908 for laminar flow over a flat plate assumes a steady, zero pressure gradient, planar or axisymmetric flow with constant density and constant properties and must be a Newtonian fluid [1]. The assumptions limit the applicability of the solutions to only simple fluid dynamics problems.

Experimental fluid dynamics tests a physical model of the aircraft and measures flow quantities of interests. Full scale flight tests are subject to the accuracy of the instruments used in the measurements but are very expensive and would only be used in the final design stages. Wind tunnel tests of a scaled model can be cheaper and more measurements can be taken since the tests are in a controlled, stationary environment, but additional uncertainty can be introduced due to wind tunnel flow quality and similitude requirements such as Reynolds number (Re) and Mach number matching.

The cheapest tool available to the fluid dynamicist is CFD. CFD uses computational resources to numerically calculate solutions to the fluid dynamics equations. Similar to analytic fluid dynamics, the equations are based on specific simplifying assumptions; however, the equations employed for CFD can have fewer assumptions drastically increasing the range of validity. In addition to the uncertainty due to the model assumptions, numerical uncertainty due to the discretization of the flow domain and governing equations, numerical methods used to solve the discretized equations, and finite computer storage are also present in the simulation results.

The analytic, experimental, and computational aspects of aerospace design are not independent of each other and all are used with increasing fidelity in aerospace design and analysis. CFD has an increasing role as computational resources become less expensive. It is important for numerical error estimation techniques to improve and further the usefulness of CFD as a fluid dynamics design tool.

1.1 Computational Fluid Dynamics

The governing equations for CFD are a set of Partial Differential Equations (PDEs) known as the Navier-Stokes (N-S) equations (see Section 4.1). These equations are derived from basic conservation principles (conservation of mass, momentum, and energy) resulting in a set of 5 equations plus an equation of state. In theory, any turbulent flow problem can be solved using the unsteady Navier-Stokes equations, called Direct Numerical Simulation (DNS). Due to limitations in computational resources, DNS solutions are limited to very specific flow regimes such as low Reynolds number flows and oblique shock-turbulent boundary layer interactions [2]. Because DNS has limited viability for aircraft design, the Navier-Stokes equations are simplified with assumptions and sub-models to reduce the computational expense of a simulation. Perhaps the most common simplified set of equations is the Reynolds Averaged Navier-Stokes (RANS) equations which employ stochastic (Reynolds averaged) terms to account for turbulent

unsteadiness always present in turbulent flows. The sub-models required to calculate the stochastic terms are empirically based on specific simplified flow regimes making certain models more or less applicable in different simulations or even different locations in the same simulation. The uncertainty in predictive capability for each model introduces an important model-form uncertainty that can be difficult and computationally expensive to estimate.

1.2 Numerical Error

Numerical errors that can be present in CFD simulations are discretization, round-off, iterative, and statistical sampling error. Discretization error and round-off error are always present in a simulation. Iterative error may be present if an iterative method is used to solve the discretized equations. Statistical sampling error may also be present if a stochastic process is used to calculate a system response quantity (i.e. time average of force). Statistical sampling error is not present in any of the simulations presented in this work because all simulations are steady and deterministic. Guidelines are given for estimation of each numerical error in Oberkampf and Roy [3].

1.2.1 Discretization Error

Discretization error is defined as the difference between the exact solution to the discrete equations u_h and the exact solution to the continuous PDEs $\tilde{u}$

$$\varepsilon_h = u_h - \tilde{u} \tag{1.1}$$

and is the largest and most difficult numerical approximation error to estimate. The source of discretization error is the discretization of the domain and the PDEs. PDEs can be solved numerically by approximating the derivatives in the governing equations using Taylor series expansions. For example, the derivative $\partial u/\partial x$ can be approximated by the second-order, central finite difference expression

$$\left(\frac{\partial u}{\partial x}\right)_i = \frac{u_{i+1} - u_{i-1}}{2\Delta x} - \left[\left(\frac{\partial^3 u}{\partial x^3}\right)_i \frac{\Delta x^2}{6} + O(\Delta x^4)\right] \tag{1.2}$$

where i is a general index for a grid (or mesh) of points distributed through the flow domain by uniform spacing Δx. The derivative at the i-th grid point is calculated using the dependent variable values stored at the i+1 and the i-1 grid point. The terms in brackets are part of the truncated terms of the Taylor series expansion and are not included in the computational solution of the PDEs. These terms are Truncation Error (TE) and are a primary source of error for a numerical simulation. The TE terms provide important information about the behavior of the discretization error as Δx becomes smaller (i.e. the grid is refined). The formal order of accuracy of the discretization scheme is the exponent on the grid spacing term for the leading term in the truncation error. In this formally second-order accurate example, the grid spacing reduced by half results in a reduction of the TE term by 1/4th if the higher-order terms designated by $O(\Delta x^4)$ are sufficiently small (i.e. the grid is in the asymptotic range).

Discretization error is related to TE and behaves in a similar manner for asymptotic solutions (at least for regular, structured meshes). There are two primary categories of discretization error estimators: recovery methods and residual-based methods. Recovery methods post process numerical solutions with variations in the

simulation such as varying the grid spacing Δx or the order of accuracy of the discretization scheme to estimate the error. Residual-based methods use an estimate of TE to modify the discretized equations. Both categories have advantages and disadvantages, but in order for any DE estimator to be accurate all solutions used to estimate the error must be in the asymptotic range, which can be very difficult to reach for even CFD simulations of moderate complexity. In addition to the requirement that solutions are in the asymptotic range, all other sources of numerical error should be at least two-orders of magnitude lower than the discretization error [4].

1.2.1.1 Richardson Extrapolation

Richardson extrapolation [11-12] is the discretization error estimator of interest in the current research and is a recovery method. Discretization error is estimated by varying the grid spacing uniformly in all directions and is given by

$$\overline{\varepsilon}_h = \frac{f_2 - f_1}{r^p - 1} \tag{1.3}$$

where f_2 and f_1 are numerical solutions with grid spacing $r\Delta x$ and Δx, respectively, r is the refinement factor and is a constant greater than one, and p is an order of accuracy. To test whether the numerical solutions are showing asymptotic behavior the observed order of accuracy is calculated

$$\hat{p} = \frac{ln\left(\frac{f_3 - f_2}{f_2 - f_1}\right)}{ln(r)} \tag{1.4}$$

where f_3, f_2, and f_1 are numerical solutions with grid spacing $r^2\Delta x$, $r\Delta x$, and Δx, respectively. Observed order of accuracy is the apparent rate of convergence of the solution and, if near the formal order, the solution is asymptotic and the error estimate is reliable. However, if the observed order of accuracy is not near the formal order, the solution is not asymptotic and the higher-order TE terms are not small. These higher order terms introduce uncertainty into the DE estimate. To account for the additional uncertainty, the absolute value of the DE estimate is multiplied by a factor of safety to create a +/- bound around the numerical solution and takes the general form

$$U = FS|\overline{\varepsilon}_h|. \tag{1.5}$$

See Chapter 3 for a more detailed discussion of Richardson extrapolation discretization error and uncertainty estimation.

1.2.2 Round-off Error

Round-off error is present due to the finite representation of data on digital computers. Accumulation of round-off error by repeated algebraic manipulation of finite numbers can make it a significant source of error for ill-conditioned systems or time-accurate simulations [3]. The effects of finite storage can be illustrated by a simple example:

$$3 * \left(\frac{1}{3}\right) = 0.9999999.$$

Data on digital computers are stored using a specific number of bits. IEEE Standard 754 [5] defines the required number of bits for a specific precision format. A

single precision calculation uses 32 bits resulting in about 7 significant digits. Double precision uses twice the number of bits and results about 15 significant digits. Other precision formats are half (3 significant digits), extended (18 significant digits), and quadruple precision (35 significant digits). The availability of precision formats is compiler dependent where double precision is the most common for scientific computing.

To estimate the round-off error in a simulation, the results of one simulation using a specific precision can be compared to the results of exactly the same simulation using a higher precision [3].

1.2.3 Iterative Error

Iterative error is present in a solution when an iterative procedure is used to solve the discretized equations and is the difference between the solution at the current iteration and the exact solution to the discretized equations. The need for an iterative method depends on the mathematical character and linearity of the equation(s) and algorithm used (e.g. explicit or implicit).

Iterative error is the exact solution to the discrete equation subtracted from the current solution:

$$\epsilon_{it}^{k} = u_h^k - u_h, \tag{1.6}$$

where ϵ_{it}^{k} is the iterative error at iteration *k*, u_h^k is the solution at iteration *k*, and u_h is the exact solution to the discrete equations. The exact solution to the discrete equations cannot be calculated exactly due to round-off error, so to estimate the iterative error the converged solution is substituted for the exact discrete solution. Iterative error estimates are only accurate when the current solution is far from the converged solution. As the current solution approaches the converged solution, the estimated iterative error approaches zero. To estimate the iterative error for the converged solution, the iterative error is extrapolated to the final iteration. Figure 1.1 shows the iterative residuals and iterative error in drag versus iteration count for a laminar flat plate case. The iterative error estimate decreases at a constant rate until the current numerical solution is near the final numerical solution. At this point the iterative error estimate quickly approaches zero. To estimate the iterative error in the final solution, the iterative error is extrapolated (black line) to the final iteration with an estimate of 10^{-17}.

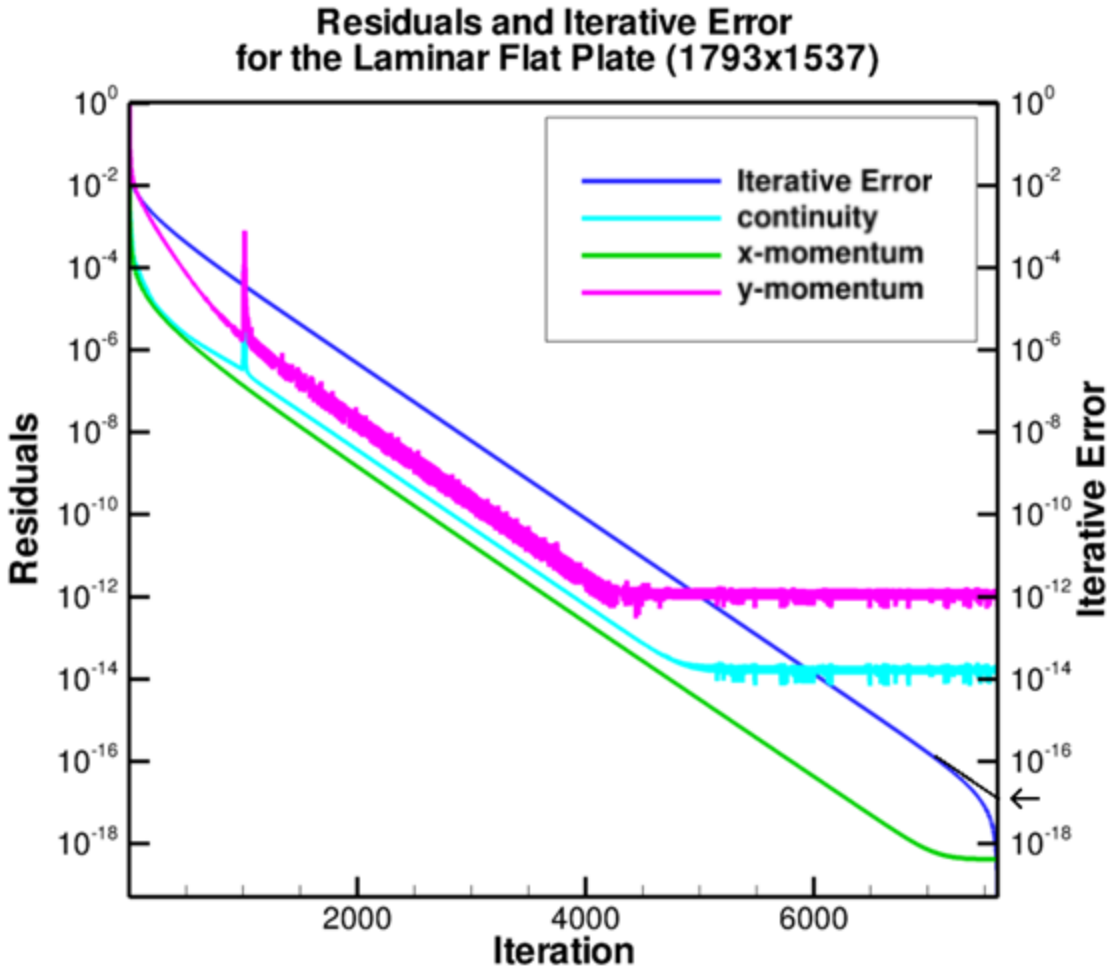


Figure 1.1. Estimated iterative error in drag of a laminar flat plate using graphical methods

2 Literature Review

A variety of discretization error estimators have been developed where some have been developed for specific discretization schemes and others are more general. Two main categories are residual-based methods and recovery methods. Residual-based methods estimate discretization error using information about the discretization scheme and the current problem and includes defect correction [6,7], error transport equations [8], and adjoint methods [9]. Residual-based methods are considerably more code intrusive than recovery methods. Recovery methods post process the solution (or solutions) to approximate the discretization error. One type of recovery method is order refinement which uses varying formal orders of accuracy to estimate discretization error and is most easily implemented in finite element methods [10]. Another type of recovery method uses a series of solutions with varying mesh spacing and can be easily implemented with any discretization scheme and is not code intrusive.

The accuracy of a discretization error estimate depends on the solution (or solutions) being in the asymptotic range which is often difficult to achieve for engineering applications. To account for the additional uncertainty in the discretization error estimate, estimates are converted to a discretization uncertainty estimate with the goal of bounding the exact solution to the PDEs with some +/- band around the numerical solution.

2.1 Discretization Error Estimation

The most commonly used extrapolation-based discretization error estimator is based off of work by Richardson [11] in 1910. The method was used to calculate a fourth-order accurate solution for a second-order finite-difference calculation of stresses in a masonry dam. The method was applied to solutions obtained on multiple mesh levels with grid doubling to estimate the true asymptotic solution. Richardson [12] later explores implementations and pitfalls of the method and demonstrates accuracy with simple applications such as the calculation of π and the natural log base e. Richardson used a refinement factor of two (i.e. grid doubling) and a formal order of accuracy of two, Richardson extrapolation has since been generalized to apply to any formal order of accuracy and refinement factor [13].

Richardson extrapolation makes the assumption that the solution is smooth and that the two solutions are asymptotic. Extrapolation can in some cases increase oscillations in the solution and can magnify iterative and round-off error. In addition, the extrapolated values are generally not conservative [13] and can result in a less accurate solution than the fine grid solution. With these considerations, Richardson extrapolation was eventually adapted as an error estimator expressed as the difference between the numerical solution and the extrapolated solution forming the base for the error and uncertainty estimators included in this study. The success of Richardson extrapolation is due to its generality because it can be applied to any scientific computation regardless of the discretization method (e.g. finite-difference, finite-volume, or finite-element) and because it is an *a-posteriori* error estimator, the method is not code intrusive and relatively easy to implement.

2.2 Discretization Uncertainty Estimation

The first uncertainty estimator developed based on Richardson extrapolation is the Grid Convergence Index (GCI) and is the absolute value of the Richardson extrapolation error estimate multiplied by a factor of safety of three. The GCI was initially proposed by Roache [14] in 1993 as a method for uniform reporting of grid convergence studies. It was pointed out by Roache [14] that the Richardson extrapolation error estimate would result in only a 50 percent conservative band around the true error and that the GCI could potentially provide a 2σ or 95 percent conservative band. The GCI was demonstrated to be a reliable uncertainty estimator, and following a study of several simple numerical solutions, Roache [15-16] made two modifications to it. The first modification included a factor of safety of 1.25 if the solution was shown to be asymptotic. The second modification used the observed order of accuracy (measure of how asymptotic the solution is) in place of the formal order of accuracy to increase the conservativeness of the uncertainty estimate. Roache [17] later modified the GCI to limit the observed order of accuracy to be no greater than the formal order of accuracy due to vanishingly small estimates for large observed orders of accuracy.

Several uncertainty estimators, either variants of the GCI or following a similar form, were developed in an attempt to create a more accurate and more reliable uncertainty estimator. One variant proposed by Cadafalch et al. [19] uses an average of the local observed orders of accuracy, and is useful for estimating error and uncertainty locally at nodes because node to node variations in observed order of accuracy can cause noisy and unreliable estimates. Stern *et al.* [20] developed two uncertainty estimators called the correction factor methods. The correction factors which were similar to the factor of safety in the GCI were derived from a one-term and two-term power series expansion. The one-term uncertainty estimator was modified by Wilson et al. [22,21] to correct some deficiencies and the two term uncertainty estimator was discontinued later when the correction factor method was applied to a hydrodynamics application [21]. Following a similar approach, the Factor of Safety method was developed by Xing and Stern [24-25]. This method calculates a factor of safety as function of the observed order of accuracy and is empirically based on data calculated from cases with known exact solutions. Taking a slightly different approach, Eca and Hoekstra [26,27] explored using a least-squares fit of several solutions in an attempt to dampen out some oscillations present in non-asymptotic solutions. Eca and Hoekstra [26] explored a variety of basis functions based on different power series expansions. The uncertainty estimator was clearly defined in Ref. 27, and this uncertainty estimator was later modified to remove some of the deficiencies in the previously published version [28].

3 Discretization Error and Uncertainty Estimation

Extrapolation-based discretization error and uncertainty estimates require a series of solutions. Richardson extrapolation and all of the uncertainty estimators based on Richardson extrapolation are applied to a set of solutions where the mesh spacing is varied using systematic grid refinement. Systematic grid refinement must be uniform and consistent [3], where uniform refinement means that the mesh is refined uniformly in all dimensional directions and consistent refinement means that the mesh quality must stay the same or improve. Assuming systematic refinement, the refinement factor r is defined as the ratio of mesh cells on the fine mesh to coarse mesh

$$r_{12} = \left(\frac{n_{c1}}{n_{c2}}\right)^{\frac{1}{d}} \tag{3.1}$$

where n_{c1} is the number of cells in the fine mesh, n_{c2} is the number of cells in the coarse mesh, and d is the grid dimension. The most common refinement factor is 2 because of simplicity. For a given fine grid on a structured mesh, every other node is removed (grid coarsening) to create the coarse mesh. For 3D grids, this results in a factor of 8 difference in the number of cells. Non-integer refinement factors can also be used; however, as the refinement factor approaches 1 the differences in the mesh size approaches zero. This results in diminishing differences between the two solutions where other forms of numerical error can affect the discretization error estimate. The minimum recommended refinement factor to prevent interference with other numerical errors is 1.1 [Ref. 17]. See Figure 3.1 for an example of grid coarsening. Non-integer refinement requires interpolation which can further introduce uncertainty in the discretization error estimate.

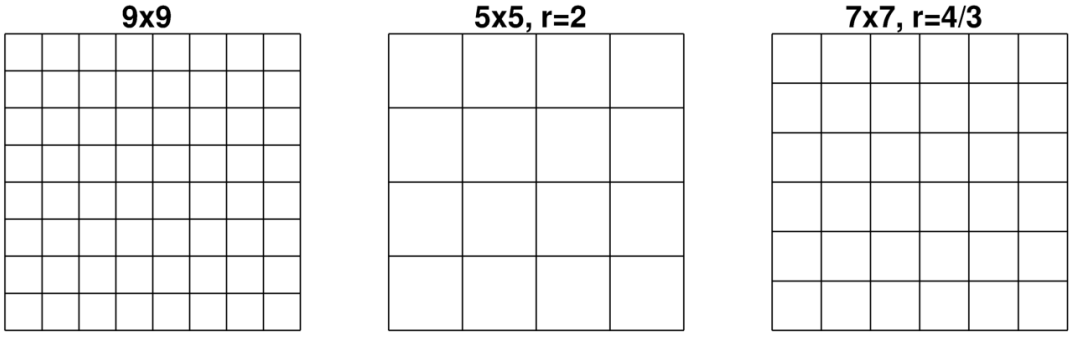


Figure 3.1. Example of grid coarsening using a refinement factor of 2 and 4/3

The reliability of all discretization error estimators require that all solutions used in the estimate are asymptotic. To estimate how asymptotic the solutions are and thus the reliability of the error estimate, an observed order of accuracy [15-16] is calculated using

$$\hat{p} = \frac{ln\left(\frac{f_3 - f_2}{f_2 - f_1}\right)}{ln(r)}. \tag{3.2}$$

If the observed order of accuracy is near the formal order of accuracy, the solutions can be considered asymptotic. Equation (3.2) is only applicable if the refinement factors between the three grids are equal. If the refinement factors differ then the observed order of accuracy is calculated using

$$\hat{p} = \frac{ln\left(\left(\frac{r_{12}^{\hat{p}} - 1}{r_{23}^{\hat{p}} - 1}\right)\left(\frac{f_3 - f_2}{f_2 - f_1}\right)\right)}{ln(r_{12})} \tag{3.3}$$

where r_{12} is the refinement factor between the fine and medium grid and r_{23} is the refinement factor between the medium and coarse grid. Equation (3.3) is solved using a fixed-point iterative method where the formal order of accuracy can be used as an initial guess.

The observed order of accuracy requires three systematically refined grids and can be calculated only for monotonically converging, $|f_2 - f_1| < |f_3 - f_2|$ and $(f_2 - f_1)(f_3 - f_2) > 0$ or monotonically diverging, $|f_2 - f_1| > |f_3 - f_2|$ and $(f_2 - f_1)(f_3 - f_2) > 0$ solutions. The order of accuracy for monotonically diverging solutions results in a negative order of accuracy and is not useful. For a 3D grid and a refinement factor of 2, this results in a factor 64 between the number of cells in the coarse grid and fine grid. The solutions used to calculate the observed order of accuracy should also be asymptotic. The large difference in number of cells in the fine mesh and the coarse mesh can make the asymptotic range difficult to achieve for all three solutions.

3.1 Richardson Extrapolation

Richardson Extrapolation is derived from a power series expansion of the numerical solution expanded about the exact solution to the PDE [11]. In the asymptotic range, only the leading term is needed:

$$\overline{f} = f_k - \alpha h_k^{p_f} \tag{3.4}$$

where $\overline{f}$ is the estimate of the exact solution, f_k is the solution on a specific grid, p_f is the formal order of accuracy, and α is an unknown coefficient. When two solutions are available on systematically-refined meshes, then the resulting two equations can be solved for the estimate of the exact solution to the PDEs

$$\overline{f} = f_1 + \frac{f_1 - f_2}{r^{p_f} - 1} \tag{3.5}$$

where f_1 and f_2 are the fine and coarse solutions, respectively, and r is the grid refinement factor. Equation (3.5) is rearranged to provide the estimated discretization error in the fine grid solution:

$$\overline{\varepsilon}_f = f_1 - \overline{f}. \tag{3.6}$$

An uncertainty estimator for the discretization error estimate using the formal order of accuracy is included in the study as a baseline comparison with other uncertainty estimators

$$U_f = \left|\overline{\varepsilon}_f\right|. \tag{3.7}$$

For asymptotic grids, the decrease in discretization error is proportional to the formal order of accuracy; however, outside of the asymptotic range the discretization error estimate is less reliable. To estimate how asymptotic the solutions are, the observed order of accuracy is calculated using Equation (3.2) or (3.3). The observed order of accuracy is used in place of the formal order of accuracy resulting in a more conservative estimate

$$\overline{\varepsilon} = \frac{f_2 - f_1}{r^{\hat{p}} - 1}. \tag{3.8}$$

An uncertainty estimator is also created using Equation (3.8) as another baseline comparison with other uncertainty estimators

$$U = |\overline{\varepsilon}|. \tag{3.9}$$

As the observed order approaches zero, $\overline{\varepsilon}$ approaches infinity. To prevent not only an infinite error estimate but also an unrealistically large error estimate, the observed order of accuracy should be limited to some small positive number, p_l. The lower limit is determined by looking at error estimates from two different manufactured solutions and an oblique shock. If treatment of oscillatory solutions is unspecified by a given method, then the order of accuracy is set to the specified lower limit. More details are given in Appendix 1. Negative observed orders of accuracy are considered oscillatory. The observed order of accuracy is also limited to the formal order of accuracy because $\hat{p} \gg p_f$ results in vanishingly-small discretization error estimates.

3.2 Grid Convergence Index and Variants

The Grid Convergence Index (GCI) was developed by Roache [14] to account for the uncertainty associated with non-asymptotic solutions. The GCI was initially developed as a method for uniform reporting of grid convergence studies using a formal order of accuracy of two ($p = 2$) and a refinement factor of two ($r = 2$) as the reference. This means that the GCI applied to grid convergence studies with different formal orders of accuracy or refinement factors than the reference values will result in an approximate uncertainty bound that would have resulted if the specified standard grid convergence study was used. For example, the difference between a fine and medium grid solution approaches zero as the refinement factor between the two grids approaches one. For discretization error reported as the difference between a fine and medium grid solution, a small refinement factor will appear to have a smaller discretization error than that resulting from a refinement factor of two; however, the GCI calculates what the difference between the two solutions would be if the reference refinement factor was used. The GCI is simply the absolute value of the Richardson extrapolation error estimate multiplied by a factor of three. The initial form of the GCI was

$$GCI = \frac{FS}{r^p - 1}\left|\frac{f_2 - f_1}{f_1}\right|. \tag{3.10}$$

The specified FS of 3 is chosen so that the normalized difference between the coarse, f_2, and fine, f_1, grid solution is recovered if the reference values for order of accuracy and refinement factor are used.

The GCI was modified later by Roache [15-16] to remove some of the deficiencies and improve generality. The first modification was the omission of the normalizing value f_1 in the denominator of Equation (3.10) due to issues when solution values are near zero. The denominator reappears later in Refs. 17 and 18, and is generally considered optional depending on user preferences. Second, Roache includes equations to determine the observed order of accuracy from solutions on three systematically refined grids. The equations are an improvement on the less exact equations given in Ref. 14. And last, Roache suggests a coefficient of 1.25 instead of 3 for cases where rigorous grid convergence studies are performed and the observed order of accuracy has been experimentally determined to be near the formal order of accuracy. In Ref. 17, Roache provides 30 simple numerical experiments using Burgers' equation to illustrate the general behavior of the GCI for varying formal orders of accuracy and refinement factors as well as to reinforce the use of coefficients of 3 and 1.25.

The GCI has evolved from a uniform reporting standard to a reliable uncertainty estimator as growing empirical evidence of its robustness and accuracy are published from the user community. A final modification to the GCI is clarified specifying the use of the observed order of accuracy in the GCI. Most modern implementations of the GCI take the general form

$$U = FS\frac{|f_2 - f_1|}{r^p - 1} \tag{3.11}$$

where *FS* is the factor of safety and p is an order of accuracy. These two values are specified based on the following conditions [14]:

if solutions on only two grids are available

$$FS = 3.0$$
$$p = p_f$$

if solutions on more than two grids are available and the observed order of accuracy is near the formal order of accuracy

$$FS = 1.25$$
$$p = \hat{p}$$

Roache specified $\hat{p} \le p_f$ (Ref. 17), but does not specify a lower limit. Roache [18] provided discussions regarding general discretization uncertainty estimation as well as specific remarks regarding the usage of the GCI and issues of non-monotone convergence.

3.2.1 Two Grid GCI

Roache's recommendations for the implementation of the GCI is not explicitly defined and requires proper judgment from the user based on a sufficient grid convergence study. Two discretization error and uncertainty estimators are clearly defined from the recommendations. The first estimator based on the first recommendation uses only two grid levels and the formal order of accuracy

$$\bar{\varepsilon}_{GCI-2G} = \frac{f_2 - f_1}{r^{p_f} - 1} \tag{3.12}$$

$$U_{GCI-2G} = 3|\bar{\varepsilon}_{GCI-2G}| \tag{3.13}$$

The two grid estimator is labeled GCI-2G. Equation (3.12) is Richardson extrapolation using the formal order of accuracy, and Equation (3.13) is the absolute value of the error estimate multiplied by a factor of safety of 3.

3.2.2 GCI with Recommendations from Oberkampf and Roy

Oberkampf and Roy [3] suggest an uncertainty estimator that explicitly defines the FS depending on the observed order of accuracy

$$\bar{\varepsilon}_{GCI-OR} = \frac{f_2 - f_1}{r^{p} - 1} \tag{3.14}$$

$$U_{GCI-OR} = FS|\bar{\varepsilon}_{GCI-OR}| \tag{3.15}$$

where values of order of accuracy and factor of safety are specified based on the observed order of accuracy:

for $\hat{p}$ within ten percent of p_f

$$FS = 1.25$$
$$p = p_f$$

for all other values

$$FS = 3.0$$
$$p = \min\left(\max(\mathrm{p_l}, \hat{p}), p_f\right)$$

The recommended lower limit, p_l, is 0.5. For cases where the observed order of accuracy is undefined, p_l should be used. This uncertainty estimator is labeled GCI-OR.

3.2.3 Global Average

A modification to the GCI proposed by Cadafalch *et al*. [19] uses an average of the local observed orders of accuracy. Only the observed order of accuracy for the monotonically converging nodes are averaged and the FS used is 1.25. First, the nodes are classified as Richardson nodes, oscillatory nodes, or converged nodes:

$$\text{Richardson nodes: } (f_3 - f_2)(f_2 - f_1) > 0$$
$$\text{Oscillatory nodes: } (f_3 - f_2)(f_2 - f_1) < 0$$

Converged nodes are nodes where the above product is below a specific tolerance and are treated the same as Richardson nodes. In their study, no converged nodes were found so the classification was omitted.

The global observed order of accuracy is calculated by averaging the observed orders of accuracy at the Richardson nodes. There is no restriction on the local observed orders of accuracy to prevent either an average above the formal order of accuracy or

below zero. To prevent global orders of accuracy outside an acceptable range, the local orders of accuracy were limit to 0.05 and p_f.

$$p = \frac{1}{N}\sum_{i=1}^{N} min\left(max(0.05, \hat{p}_i), p_f\right) \tag{3.16}$$

Next, the global order of accuracy is used to calculate the discretization error estimate and uncertainty at every location using

$$\overline{\varepsilon}_{GCI-glb} = \frac{f_2 - f_1}{r^p - 1} \tag{3.17}$$

$$U_{GCI-glb} = 1.25|\overline{\varepsilon}_{GCI-glb}| \tag{3.18}$$

In Ref. 19, oscillatory data points and boundary data points were excluded; however, all data points are included for consistency.

3.3 Correction Factor Method

The correction factor method was initially proposed by Stern et al. [20], and later modified by Wilson et al. [22] after criticism from Roache [23]. The modification changed the asymptotic behavior of the correction factor from 1.0 to 1.1, because of 50 percent conservativeness when 1.0 is used. Another modification makes the correction factor symmetric about the formal order of accuracy.

The current implementation of the Correction Factor (CF) method takes the form

$$\overline{\varepsilon}_{CF} = \frac{f_2 - f_1}{r^{\hat{p}} - 1} \tag{3.19}$$

$$U_{CF} = \begin{cases} [9.6(1-CF)^2 + 1.1]|\overline{\varepsilon}|, & 0.875 < CF < 1.125 \\ [2|1-CF| + 1]|\overline{\varepsilon}|, & 0 < CF \leq 0.875 \; and \; CF \geq 1.125 \end{cases} \tag{3.20}$$

where

$$CF = \frac{r^{\hat{p}} - 1}{r^{p_f} - 1} \tag{3.21}$$

Diverging data points are excluded from the uncertainty estimates in Ref. 20; however, all diverging and oscillatory data points were limited to the specified lower limit, p_l, as well as all converging data points below the lower limit.

3.4 Factor of Safety Method

The Factor of Safety method was proposed by Xing and Stern [24] and is similar to the Correction Factor method in that FS is a function of observed order of accuracy. The primary difference between the two methods is that the Factor of Safety method has two coefficients which were calibrated using a sample set of 17 different multidimensional fluid dynamic cases and structural cases. The Factor of Safety method was later modified by Xing and Stern [25] to have three coefficients and removing a singularity at twice the formal order of accuracy. No conditions are specified for limiting $\hat{p}$ by a lower or upper value. For consistency, the same lower limit is applied to the factor of safety method, $p = max(p_l, \hat{p})$. No upper limit is implemented because the uncertainty estimator was

designed for observed orders of accuracy greater than the formal order. The factor of safety method is defined as

$$\overline{\varepsilon}_{FS} = \frac{f_2 - f_1}{r^p - 1} \tag{3.22}$$

$$U_{FS} = \begin{cases} [FS_1 P + FS_0(1-P)]|\overline{\varepsilon}_{FS}|, & 0 < P \leq 1 \\ [FS_1 P + FS_2(P-1)]|\overline{\varepsilon}_{FS}|, & P > 1 \end{cases} \tag{3.23}$$

where $FS_0 = 2.45$, $FS_1 = 1.6$, and $FS_2 = 14.8$ and $P = \hat{p}/p_f$.

3.5 Least-Squares Method

Due to non-asymptotic solutions, error estimates can be noisy, Eca and Hoekstra [26] explored a least squares approach using power series expansions as the basis functions to estimate discretization error and reduce noise. Different one and two term power series expansions with unknown or assumed exponents are considered. The power series expansion that is used to derive Richardson extrapolation requires at least four grid levels to solve for the discretization error, exponent, and coefficient. Other power series expansions assume an exponent such as a one-term expansion with a fixed exponent of two, or a two-term expansion with fixed exponents of one and two.

Using Equation (3.4) an error function is created:

$$S(\bar{f}, \alpha, \tilde{p}) = \left[\sum_{k=1}^{n_g} \left[f_k - \left(\bar{f} + \alpha h_k^{\tilde{p}} \right) \right]^2 \right]^{\frac{1}{2}} \tag{3.24}$$

where $\tilde{p}$ is the least squares observed order of accuracy, and f_k is the k-th solution on a grid with spacing h_k. The derivatives of S with respect to $\overline{f}$, α, and $\tilde{p}$ are set to zero resulting in a set of three equations solved using the false-position root finding method:

$$\alpha = \frac{n_g \sum_{k=1}^{n_g} f_k h_k^{\tilde{p}} - \left(\sum_{k=1}^{n_g} f_k \right) \left(\sum_{k=1}^{n_g} h_k^{\tilde{p}} \right)}{n_g \sum_{k=1}^{n_g} h_k^{2\tilde{p}} - \left(\sum_{k=1}^{n_g} h_k^{\tilde{p}} \right)^2} \tag{3.25}$$

$$\overline{f} = \frac{\sum_{k=1}^{n_g} f_k - \alpha \sum_{k=1}^{n_g} h_k^{\tilde{p}}}{n_g} \tag{3.26}$$

$$\sum_{k=1}^{n_g} f_k h_k^{\tilde{p}} \, ln(h_k) - \overline{f} \sum_{k=1}^{n_g} h_k^{\tilde{p}} \, ln(h_k) - \alpha \sum_{k=1}^{n_g} h_k^{2\tilde{p}} \, ln(h_k) = 0. \tag{3.27}$$

The least squares approach requires at least four solutions (three solutions would reproduce the same results as Equation (3.8)). The discretization error and uncertainty estimator, labeled LSQ-09, are subject to different conditions depending on the value of the $\tilde{p}$ for $p_f = 2$:

$0.95 \leq \tilde{p} \leq 2.05$	$U_{LSQ-09} = 1.25\varepsilon_{LSQ} + U_s$
$0 < \tilde{p} \leq 0.95$	$U_{LSQ-09} = \min(1.25\varepsilon_{LSQ} + U_s, 1.25\Delta_M)$
$\tilde{p} \geq 2.05$	$U_{LSQ-09} = \max(1.25\varepsilon_{LSQ} + U_s, 1.25\Delta_M)$

else $U_{LSQ-09} = 3\Delta_M$

where $\varepsilon_{LSQ} = f_1 - \overline{f}$, U_s is the RMS of the fit,

$$U_s = \sqrt{\sum_{k=1}^{n_g} \frac{\left(f_k - \left(\overline{f} + \alpha h_k^{\tilde{p}}\right)\right)^2}{n_g - 3}} \tag{3.28}$$

and

$$\Delta_M = max\left(\left|f_i - f_j\right|\right) \quad 1 \le i \le n_g, 1 \le j \le n_g \tag{3.29}$$

The least squares method was modified in Ref. 28 to correct some of the deficiencies in the LSQ-09 method. This method, labeled LSQ-10, estimates the discretization error using two additional error functions based on fixed-exponent power series expansions. For cases where the least squares observed order of accuracy is greater than the formal order of accuracy, $\varepsilon_{LSQ}^2 = \alpha h_k^2$ is used, and if the least squares observed order of accuracy is not near the formal order of accuracy, then $\varepsilon_{LSQ}^{12} = \alpha_1 h_k + \alpha_2 h_k^2$ is used. (Note that the superscript n in ε_{LSQ}^n is a label and not an exponent). If the observed order of accuracy does not fit the above conditions then $\varepsilon_{\Delta_M} = \frac{\Delta_M}{r^{p_m}-1}$ is used, where $p_m = 1$ (Ref. 28). Uncertainty estimation is outlined in Ref. 28 as follows

$0.95 \le \tilde{p} \le 2.05$ $\quad U_{LSQ-10} = 1.25\varepsilon_{LSQ} + U_s$

$0 < \tilde{p} \le 0.95$ $\quad U_{LSQ-10} = \min\left(1.25\varepsilon_{LSQ} + U_s, 3\varepsilon_{LSQ}^{12} + U_s^{12}\right)$

$\tilde{p} \ge 2.05$ $\quad U_{LSQ-10} = \max\left(1.25\varepsilon_{LSQ} + U_s, 3\varepsilon_{LSQ}^{2} + U_s^{2}\right)$

else $\quad U_{LSQ-10} = 3\varepsilon_{\Delta_M}$

The equations used to calculate ε_{LSQ}^{12} are

$$\overline{f} n_g + \alpha_1 \sum_{k=1}^{n_g} h_k + \alpha_2 \sum_{k=1}^{n_g} h_k^2 = \sum_{k=1}^{n_g} f_k \tag{3.30}$$

$$\overline{f} \sum_{k=1}^{n_g} h_k + \alpha_1 \sum_{k=1}^{n_g} h_k^2 + \alpha_2 \sum_{k=1}^{n_g} h_k^3 = \sum_{k=1}^{n_g} f_k h_k \tag{3.31}$$

$$\overline{f} \sum_{k=1}^{n_g} h_k^2 + \alpha_1 \sum_{k=1}^{n_g} h_k^3 + \alpha_2 \sum_{k=1}^{n_g} h_k^4 = \sum_{k=1}^{n_g} f_k h_k^2 \tag{3.32}$$

$$U_s^{12} = \sqrt{\frac{\sum_{k=1}^{n_g} \left(f_k - \left(\overline{f} + \alpha_1 h_k + \alpha_2 h_k^2\right)\right)^2}{n_g - 3}} \tag{3.33}$$

and ε_{LSQ}^2

$$\overline{f} n_g + \alpha \sum_{k=1}^{n_g} h_k = \sum_{k=1}^{n_g} f_k \tag{3.34}$$

$$\overline{f} \sum_{k=1}^{n_g} h_k^2 + \alpha \sum_{k=1}^{n_g} h_k^4 = \sum_{k=1}^{n_g} f_k h_k^2 \tag{3.35}$$

$$U_s^2 = \sqrt{\frac{\sum_{k=1}^{n_g} \left(f_k - \left(\overline{f} + \alpha h_k^2 \right) \right)^2}{n_g - 2}} \tag{3.36}$$

For both the LSQ-09 and LSQ-10, the error estimate for situations requiring the use of the *min* or *max* function corresponds to the resulting uncertainty calculation. E.g., for $\tilde{p} = 2.5$ and $\left(1.25\varepsilon_{LSQ} + U_s\right) < \left(3\varepsilon_{LSQ}^2 + U_s^2\right)$, the error estimate is ε_{LSQ}^2.

4 Flow Equations

The four governing equations for two-dimensional flow are the Navier-Stokes and Euler equations with the assumption of a perfect gas. The equations are based on the conservation of mass, momentum, and energy, and the solution variables are density (ρ), x-velocity (u), y-velocity (v), and pressure (p). The Navier-Stokes equations are described in Section 4.1. The Euler equations are the Navier-Stokes equations with viscous terms removed.

4.1 Two-dimensional, Navier-Stokes Equations

The 2D, steady state, Favre-averaged Navier-Stokes equations [31] can be written as

$$\frac{\partial(\rho u)}{\partial x}+\frac{\partial(\rho v)}{\partial y}=0 \tag{4.1}$$

$$\frac{\partial(\rho u^2+p-t_{xx}-\tau_{xx})}{\partial x}+\frac{\partial\left(\rho uv-t_{xy}-\tau_{xy}\right)}{\partial y}=0 \tag{4.2}$$

$$\frac{\partial\left(\rho vu-t_{xy}-\tau_{xy}\right)}{\partial x}+\frac{\partial\left(\rho v^2+p-t_{yy}-\tau_{yy}\right)}{\partial y}=0 \tag{4.3}$$

$$\begin{aligned}&\frac{\partial\left(\rho uh_t-u(t_{xx}+\tau_{xx})-v\left(t_{xy}+\tau_{xy}\right)+q_{L_x}+q_{T_x}\right)}{\partial x}\\&+\frac{\partial\left(\rho vh_t-u\left(t_{xy}+\tau_{xy}\right)-v\left(t_{yy}+\tau_{yy}\right)+q_{L_y}+q_{T_y}\right)}{\partial y}=0\end{aligned} \tag{4.4}$$

The laminar stress tensor t_{ij} is given by

$$\begin{aligned}t_{xx}&=\frac{2}{3}\mu\left(2\frac{\partial u}{\partial x}-\frac{\partial v}{\partial y}\right),\\t_{yy}&=\frac{2}{3}\mu\left(2\frac{\partial v}{\partial y}-\frac{\partial u}{\partial x}\right),\\t_{xy}&=\mu\left(\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\right),\end{aligned} \tag{4.5}$$

and the turbulent stress tensor τ_{ij} is given by

$$\begin{aligned}\tau_{xx}&=\frac{2}{3}\mu_T\left(2\frac{\partial u}{\partial x}-\frac{\partial v}{\partial y}\right),\\\tau_{yy}&=\frac{2}{3}\mu_T\left(2\frac{\partial v}{\partial y}-\frac{\partial u}{\partial x}\right),\\\tau_{xy}&=\mu_T\left(\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\right).\end{aligned} \tag{4.6}$$

The turbulent and laminar heat flux terms are

$$\begin{aligned} q_{T_x} &= -\frac{\mu_T}{Pr_T} c_p \frac{\partial T}{\partial x}, \\ q_{T_y} &= -\frac{\mu_T}{Pr_T} c_p \frac{\partial T}{\partial y}, \\ q_{L_x} &= -\frac{\mu}{Pr} c_p \frac{\partial T}{\partial x}, \\ q_{L_y} &= -\frac{\mu}{Pr} c_p \frac{\partial T}{\partial y}, \end{aligned} \tag{4.7}$$

and the total energy and enthalpy are

$$e_t = e + \frac{1}{2}(u^2 + v^2) \tag{4.8}$$

$$h_t = \ e_t + \frac{P}{\rho} \tag{4.9}$$

where

$$e = nRT + \ h_f \tag{4.10}$$

$$h = (n+1)RT + \ h_f \tag{4.11}$$

The perfect gas equation of state is assumed as

$$P = \rho RT \tag{4.12}$$

and the heat of formation and excited energy mode parameter can be calculated as

$$h_f = \ h_{ref} - (n+1)RT_{ref} \quad and \quad n = \frac{c_v}{R} \tag{4.13}$$

where $h_{ref} = 0$, $T_{ref} = 298$K, and $n = 5/2$ are used.

The turbulent equations can be reduced to laminar equations by setting turbulent stress tensor τ_{ij} and the turbulent viscosity μ_T to zero.

4.1.1 Turbulence Modeling: k-ω SST-V

Additional equations to model the turbulence are required. The model used for this research is the Menter SST two-equation turbulence model with vorticity source term [32,33]:

$$\frac{\partial(\rho k)}{\partial t} + \frac{\partial(\rho u_j k)}{\partial x_j} = P - \beta^* \rho \omega k + \frac{\partial}{\partial x_j}\left[(\mu + \sigma_k \mu_t)\frac{\partial k}{\partial x_j}\right] \tag{4.14}$$

$$\frac{\partial(\rho\omega)}{\partial t}+\frac{\partial\left(\rho u_j\omega\right)}{\partial x_j}=\frac{\gamma}{\nu_t}P-\beta\rho\omega^2+\frac{\partial}{\partial x_j}\left[(\mu+\sigma_\omega\mu_t)\frac{\partial\omega}{\partial x_j}\right] +2(1-F_1)\frac{\rho\sigma_{\omega2}}{\omega}\frac{\partial k}{\partial x_j}\frac{\partial\omega}{\partial x_j} \tag{4.15}$$

where ρ is density, $\nu_t=\mu_t/\rho$ is the turbulent kinematic viscosity, μ is the molecular dynamic viscosity, d is the distance to the nearest wall, and ω is the vorticity magnitude. Each of the constants is a blend of an inner (1) and outer (2) constant, represented by ϕ_1 and ϕ_2:

$$P=\mu_t\omega^2-\frac{2}{3}\rho k\delta_{ij}\frac{\partial u_i}{\partial x_j},\qquad \mu_t=\frac{\rho a_1 k}{\max(a_1\omega,\omega F_2)},$$

$$CD_{k\omega}=\max\left(2\rho\sigma_{\omega2}\frac{1}{\omega}\frac{\partial k}{\partial x_j}\frac{\partial\omega}{\partial x_j},10^{-20}\right),\qquad \phi=F_1\phi_1+(1-F_1)\phi_2,$$

$$arg_1=\min\left[max\left(\frac{\sqrt{k}}{\beta^*\omega d},\frac{500\nu}{d^2\omega}\right),\frac{4\rho\sigma_{\omega2}k}{CD_{k\omega}d^2}\right],\qquad F_1=\tanh(arg_1^4),$$

$$arg_2=\max\left(2\frac{\sqrt{k}}{\beta^*\omega d},\frac{500\nu}{d^2\omega}\right),\qquad F_2=\tanh(arg_2^2).$$

The production term in the k-equation is limited using $min(P,20\beta^*\rho\omega k)$, and the constants are

$$\sigma_{k1}=0.85,\quad \sigma_{\omega1}=0.5,\quad \beta^*=0.09,$$
$$\sigma_{k2}=1.0,\quad \sigma_{\omega2}=0.856,\quad a_1=0.31,$$
$$\beta_1=0.075,\quad \beta_2=0.0828,\quad \kappa=0.41,$$
$$\gamma=F_1\gamma_1+(1-F_1)\gamma_2,$$
$$\gamma_1=\frac{\beta_1}{\beta^*}-\frac{\sigma_{\omega1}\kappa^2}{\sqrt{\beta^*}},\qquad \gamma_2=\frac{\beta_2}{\beta^*}-\frac{\sigma_{\omega2}\kappa^2}{\sqrt{\beta^*}}.$$

5 Exact Solutions

5.1 Manufactured Solutions

Exact solutions can be created by modifying the governing equations with a source term that is calculated from a solution chosen *a priori*. This exact process is used for code verification and is called the Method of Manufactured Solutions [34]. The chosen solution does not need to physically realistic but should be smooth and be compatible with the boundary conditions of the simulation.

Two different manufactured solutions are used in the study. Both solutions are chosen to have the same form except one is supersonic and the other is subsonic. The general form of the equations is

$$\begin{aligned} \rho(x,y) &= a_{\rho 0} + a_{\rho x}\, sin\left(\frac{s_{\rho x}\pi x}{L}\right) - a_{\rho y}\, cos\left(\frac{s_{\rho y}\pi y}{L}\right) \\ u(x,y) &= a_{u0} + a_{ux}\, sin\left(\frac{s_{ux}\pi x}{L}\right) - a_{uy}\, cos\left(\frac{s_{uy}\pi y}{L}\right) \\ v(x,y) &= a_{v0} + a_{vx}\, sin\left(\frac{s_{vx}\pi x}{L}\right) - a_{vy}\, cos\left(\frac{s_{vy}\pi y}{L}\right) \\ p(x,y) &= a_{p0} + a_{px}\, sin\left(\frac{s_{px}\pi x}{L}\right) - a_{py}\, cos\left(\frac{s_{py}\pi y}{L}\right) \end{aligned} \tag{5.1}$$

where L is a reference length chosen to be 1. The 2D Euler equations then operate on Equation (5.1) to create source terms which are added to the right-hand-side of the discretized governing equations equations. Since the exact solution is known, the exact discretization error can be calculated.

Both the supersonic manufactured solution and the subsonic manufactured solution are solved on the same curvilinear mesh family where the finest mesh is $513x513$. Two sets of grids are created for error estimation. The first set of meshes consists of the finest mesh coarsen by a refinement factor of 2 to the coarsest of $17x17$. The second set of meshes consists of each mesh in the first set of meshes plus the first set of meshes coarsened by a factor of 4/3. For example, the first set uses $513x513$, $257x257$, and $129x129$ with a refinement factor of 4 from the finest to coarsest mesh. The second set of meshes uses $513x513$, $385x385$, and $257x257$ with a refinement factor of 2 from the finest to coarsest mesh. The two mesh sets are included to study the effect of the coarser, less asymptotic solutions on discretization error and uncertainty estimation. The mesh combinations for each mesh set are shown in Table 5.1. The bolded mesh is the mesh for the corresponding error estimate and the fourth mesh is included for use with the least squares methods only. If a fourth mesh is not included, an estimate was not calculated for the least squares methods.

Table 5.1. Manufactured solution mesh combinations

Mesh family 1 combinations				Mesh family 2 combinations			
513x513	257x257	129x129	65x65	**513x513**	385x385	257x257	193x193
257x257	129x129	65x65	33x33	**257x257**	193x193	129x129	97x97
129x129	65x65	33x33	17x17	**129x129**	97x97	65x65	49x49
65x65	33x33	17x17		**65x65**	49x49	33x33	25x25
				33x33	25x25	17x17	13x13

5.1.1 Supersonic Solution

The supersonic manufactured solution is specified using the coefficients shown in Table 5.2. Contour plots of the exact solution are shown in Figure 5.1.

Table 5.2. Summary of coefficients for the supersonic manufactured solution

	a_0	a_x	a_y	s_x	s_y
$\rho(x,y)$	1.0	0.15	-0.1	1.0	0.5
$u(x,y)$	800.0	50.0	-30.0	1.5	0.6
$v(x,y)$	800.0	-75.0	40.0	0.5	0.6
$p(x,y)$	1.0E5	2.0E4	5.0E4	2.0	1.0

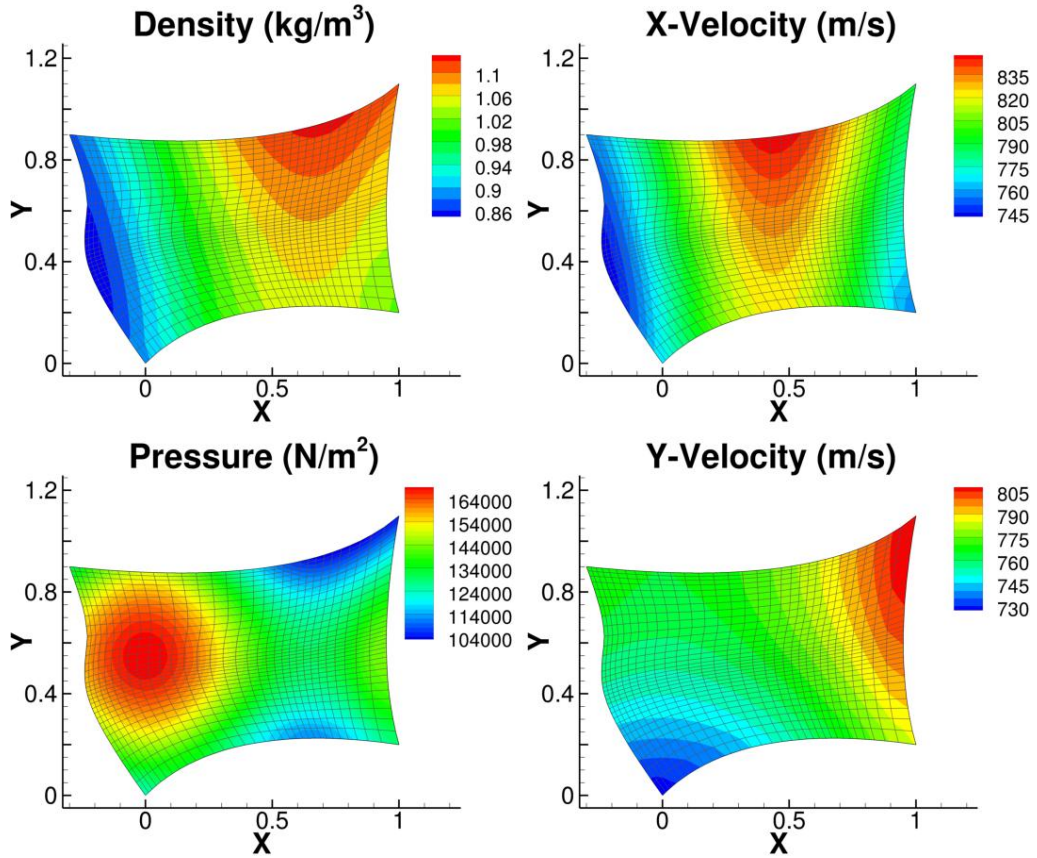


Figure 5.1. Supersonic manufactured solution primitive variables with 33x33 mesh

5.1.2 Subsonic Solution

The supersonic manufactured solution is specified using the coefficients shown in Table 5.3. Contour plots of the exact solution are shown in Figure 5.2.

Table 5.3. Summary of coefficients for the subsonic manufactured solution

	a_0	a_x	a_y	s_x	s_y
$\rho(x,y)$	1.0	0.15	0.1	1.0	0.5
$u(x,y)$	-70.0	-25.0	-30.0	2.5	0.6
$v(x,y)$	-90.0	-15.0	25.0	4.0	1.0
$p(x,y)$	1.0E5	2.0E4	5.0E4	2.0	1.0

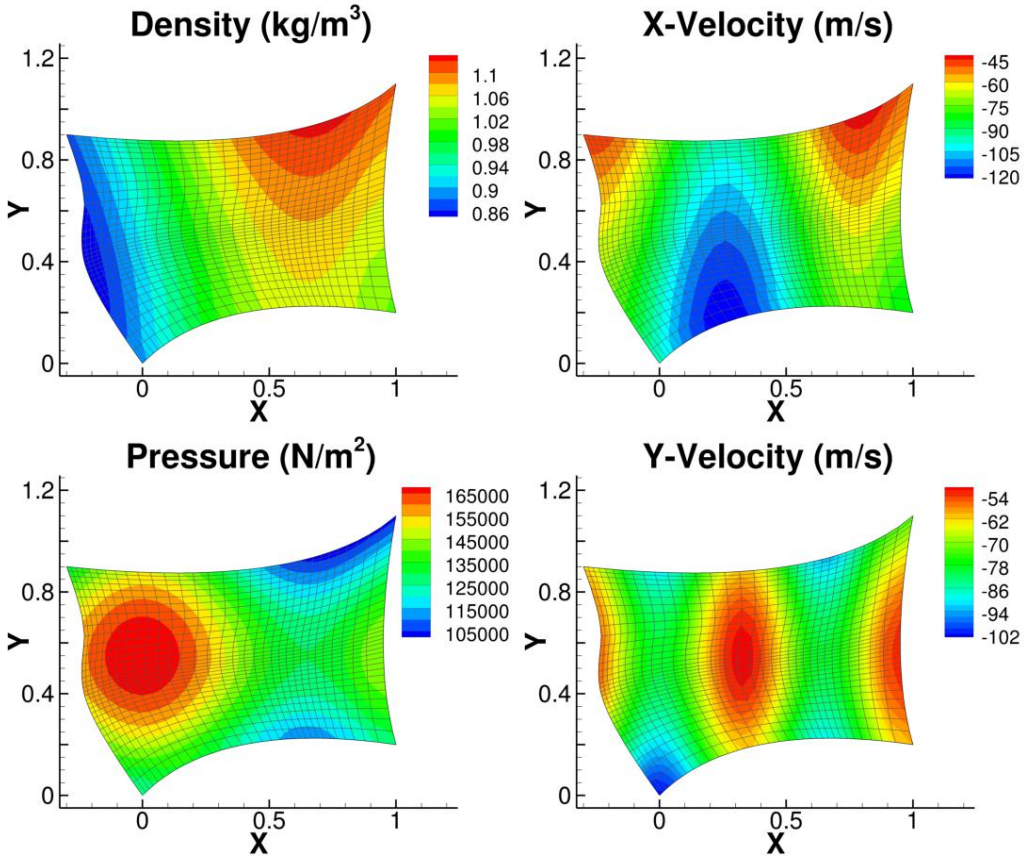


Figure 5.2. Subsonic manufactured solution primitive variables with 33x33 mesh

5.2 Ringleb's Flow

Ringleb's flow is an exact solution to the 2D Euler equations and consists of flow around a 180 degree turn [29]. The flow can be supersonic, subsonic, or a both depending on the solution domain chosen. For this study, a supersonic only domain was chosen. Ringleb's flow is governed by the stream function

$$\psi = \frac{1}{q} sin(\theta). \tag{5.2}$$

The inverse of the stream function is defined as

$$k = \frac{1}{\psi}. \tag{5.3}$$

The flow direction is defined as

$$\theta = sin^{-1}\left(\frac{q}{k}\right) \tag{5.4}$$

where q is the normalized velocity of the flow and theta is the flow angle. The spatial locations are functions of q and k

$$x = \frac{1}{2h}\left(\frac{1}{q^2} - \frac{2}{k^2}\right) + \frac{J}{2} \tag{5.5}$$

$$y = \pm\frac{1}{qhk}\sqrt{1 - \left(\frac{q^2}{k^2}\right)} \tag{5.6}$$

where

$$h = c^{\frac{2}{\gamma-1}} \tag{5.7}$$

$$J = \frac{1}{c} + \frac{1}{3c^3} + \frac{1}{5c^5} - \frac{1}{2}ln\left(\frac{1+c}{1-c}\right) \tag{5.8}$$

$$c = \sqrt{1 - \frac{\gamma-1}{2}q^2}. \tag{5.9}$$

The Mach number is given by

$$M = \sqrt{\frac{q^2}{1 - \frac{\gamma-1}{2}q^2}} \tag{5.10}$$

The variable q varies along the streamline defined by k from q_{min} to k, where $k_{max} < \sqrt{5}$ for $\gamma = 1.4$. The flow field is defined in terms of k and q which define the x and y grid point locations. From the Mach number and the flow angle, the primitive variables are calculated using isentropic relations for air at *P = 100,000 Pa, T = 300.0 K*. For this analysis, $1.1 < k < 1.6$ and the minimum value of $q = 1.05$. The resulting Mach range is $1.19 < M < 2.29$. See Figure 5.3 for the exact solution for each primitive variable.

Two mesh families are chosen similar to those for the manufactured solution, see Table 5.4. The inlet boundary condition for Ringleb's specifies the exact flow direction and flow velocity. The streamline boundaries corresponding to k_{min} and k_{max} are modeled with a slip wall boundary condition, and the outflow is modeled as a supersonic outflow where all variables are extrapolated from the interior of the domain.

Table 5.4. Ringleb's flow mesh combinations

Mesh family 1 combinations				Mesh family 2 combinations			
257x257	129x129	65x65	33x33	**257x257**	193x193	129x129	97x97
129x129	65x65	33x33	17x17	**129x129**	97x97	65x65	49x49
65x65	33x33	17x17		**65x65**	49x49	33x33	25x25
				33x33	25x25	17x17	13x13

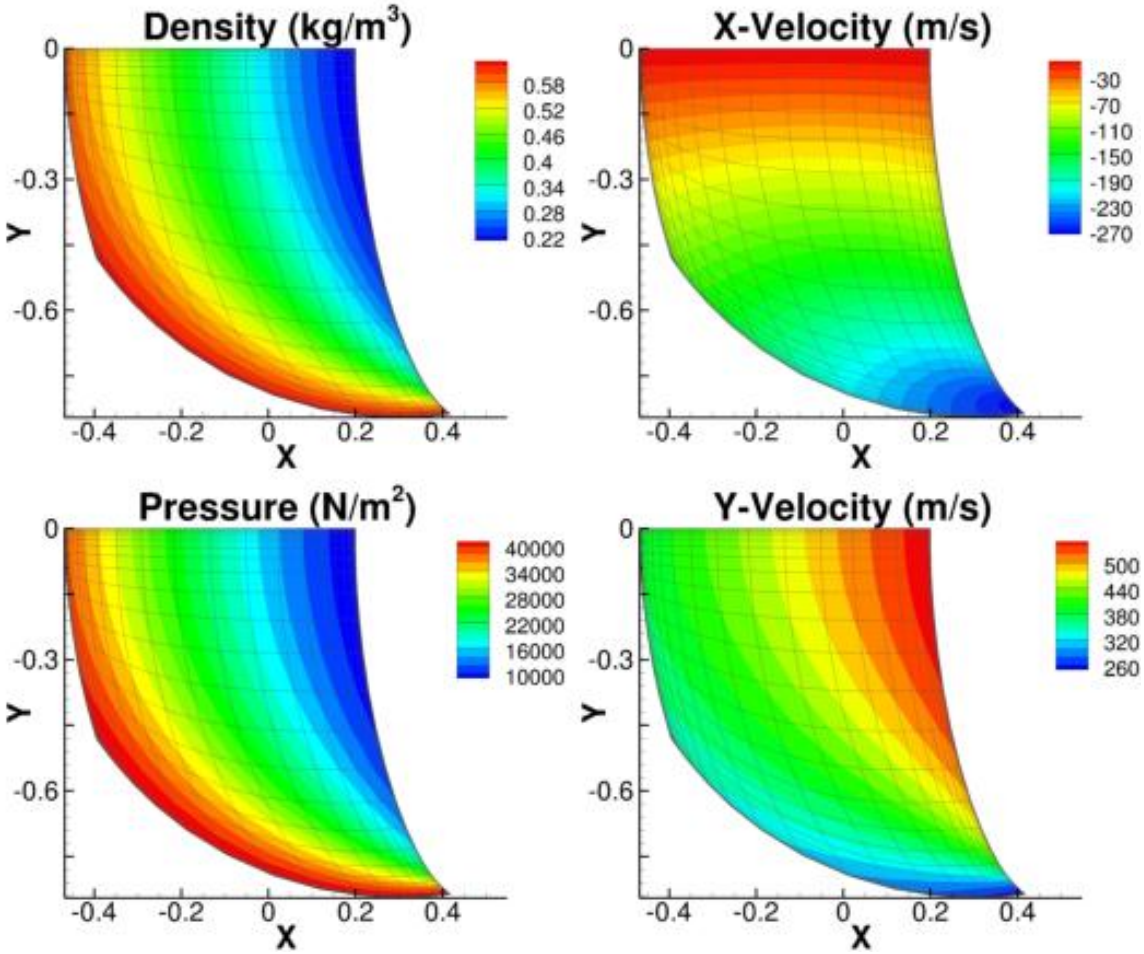


Figure 5.3. Ringleb's flow primitive variables with 33x33 mesh

5.3 Supersonic Vortex Flow

Supersonic vortex flow [30] consists of a flow around a 90 degrees annulus. The flow field is defined as a function of variables at the inner radius of the annulus denoted by the subscript *i*. The inner radius, R_i, is 2.0 m, the outer radius is 3.0 m, the inner density, ρ_i, is 1.0 kg/m^3, and the inner Mach number, M_i, is 2.0. The primitive variables are calculated using

$$\rho(r) = \rho_i \left(1 + \frac{\gamma - 1}{2} M_i^2 \left(1 - \frac{R_i^2}{r^2}\right)\right)^{\frac{1}{\gamma - 1}} \tag{5.11}$$

$$u(y,r) = \frac{yU}{r} \tag{5.12}$$

$$v(x,r) = -\frac{xU}{r} \tag{5.13}$$

$$P(\rho) = \frac{\rho^{\gamma}}{\gamma} \tag{5.14}$$

where

$$U_i = M_i \rho_i^{\frac{\gamma-1}{2}} \tag{5.15}$$

$$U(r) = \frac{U_i R_i}{r} \tag{5.16}$$

The finest mesh created is 513x257 and coarsened by factors of two to create the coarsest mesh of 17x9. Like the other cases, estimates are calculated on two different sets of grids with different refinement factors. Table 5.5 shows the two different sets of grids and how they are combined to estimate the discretization error and uncertainty.

Table 5.5. Supersonic vortex flow mesh combinations

Mesh family 1 combinations				Mesh family 2 combinations			
513x257	257x129	129x65	65x33	**513x257**	385x193	257x129	193x97
257x129	129x65	65x33	33x17	**257x129**	193x97	129x65	97x49
129x65	65x33	33x17	17x9	**129x65**	97x49	65x33	49x25
65x33	33x17	17x9		65x33	49x25	33x17	25x13
				33x17	25x13	17x9	

The boundary conditions for the supersonic vortex flow are the same as for Ringleb's flow. The inflow is a Dirichlet boundary condition with all flow variables specified. The streamlines of the flow are modeled as a slip wall and the outflow is a supersonic outflow boundary condition where all flow variables are extrapolated from the interior.

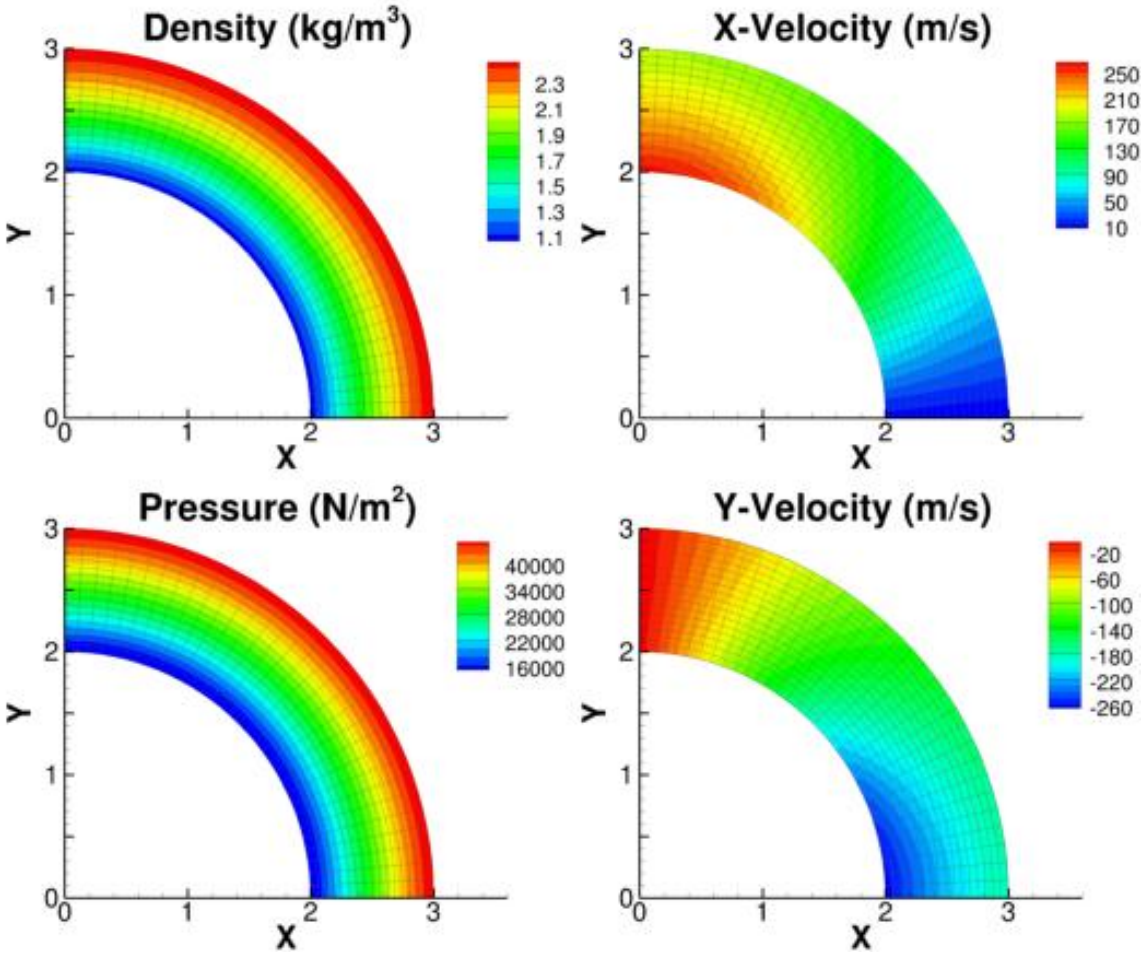


Figure 5.4. Supersonic vortex flow primitive variables with 33x17 mesh

6 Discretization Method and Code Verification

6.1 Discretization of the Euler Equations

An in-house flow solver was developed to solve the 2D Euler equations using a finite-volume discretization scheme. The flow solver implements Roe's upwind scheme [35] and is formally second-order accurate using MUSCL extrapolation [36].

6.1.1 Finite-Volume method

The two-dimensional Euler equations are simplified from the Navier-Stokes equations given in Chapter 4 where all of the viscous terms are set to zero: t_{xx}, t_{xy}, t_{yy}, τ_{xx}, τ_{xy}, and τ_{yy}. Also all heat transfer terms are set to zero: q_{L_x}, q_{L_y}, q_{T_x}, and q_{T_y}. The 2D Euler equations in generalized coordinates are written in conservative form as

$$\frac{\partial U_1}{\partial t} + \frac{\partial F_1}{\partial \xi} + \frac{\partial G_1}{\partial \eta} = S_1 \tag{6.1}$$

where the vector of conserved variables and the vector of fluxes are given by

$$U_1 = \frac{1}{J}\begin{bmatrix} \rho \\ \rho U \\ \rho V \\ \rho e_t \end{bmatrix} \tag{6.2}$$

$$F_1 = \frac{1}{J}\begin{bmatrix} \rho U \\ \rho U^2 + p\xi_x \\ \rho UV + p\xi_y \\ \rho U h_t \end{bmatrix} \tag{6.3}$$

$$G_1 = \frac{1}{J}\begin{bmatrix} \rho V \\ \rho UV + p\eta_x \\ \rho V^2 + p\eta_y \\ \rho V h_t \end{bmatrix} \tag{6.4}$$

and $S_1 = S/J$ where S is a source term which can be used to modify the governing equations. The computational coordinate system consists of equally spaced data with mesh spacing of one in both the ξ and η directions. Transformation metrics are used to transform a general mesh to the computational mesh assuming $\Delta\xi = \Delta\eta = 1$:

$$x_\xi = \frac{x_{i+1,j} - x_{i-1,j}}{2} \tag{6.5}$$

$$x_\eta = \frac{x_{i,j+1} - x_{i,j-1}}{2} \tag{6.6}$$

$$y_\xi = \frac{y_{i+1,j} - y_{i-1,j}}{2} \tag{6.7}$$

$$y_\eta = \frac{y_{i,j+1} - y_{i,j-1}}{2} \tag{6.8}$$

$$J = x_\xi y_\eta - x_\eta y_\xi \tag{6.9}$$

$$\xi_x = y_\eta / J \tag{6.10}$$

$$\xi_y = -x_\eta / J \tag{6.11}$$

$$\eta_x = -y_\xi / J \tag{6.12}$$

$$\eta_y = x_\xi / J \tag{6.13}$$

The transformation metrics are also used to transform the solution variables to the computational mesh. The contravarient velocities (velocities in transformed coordinate system and do not have the same magnitude as the physical velocities) are calculated using

$$\widehat{U} = \xi_x u + \xi_y v \tag{6.14}$$

$$\widehat{V} = \eta_x u + \eta_y v \tag{6.15}$$

6.1.2 Roe's Upwind Scheme

The governing equations are discretized using Roe's flux difference splitting scheme [35]. Roe's scheme calculates the flux at an interface using

$$\boldsymbol{f}_{i+\frac{1}{2}} = \frac{1}{2}\left(\boldsymbol{f}^L_{i+\frac{1}{2}} + \boldsymbol{f}^R_{i+\frac{1}{2}}\right) - \frac{1}{2}\sum_{j=1}^{n} \overline{\overline{\lambda_j}}\,\delta w_j \overline{\overline{r_j}}. \tag{6.16}$$

The fluxes at the left ($\boldsymbol{f}^L$) and right ($\boldsymbol{f}^R$) of the interface are calculated using the flux functions defined in Equation (6.3) or (6.4) depending on the interface. The eigenvalues $\overline{\overline{\lambda_j}}$ for the flux in the ξ-directions are

$$\overline{\overline{\boldsymbol{\lambda}}} = \begin{bmatrix} \overline{\overline{u}}\xi_x + \overline{\overline{v}}\xi_y \\ \overline{\overline{u}}\xi_x + \overline{\overline{v}}\xi_y \\ \overline{\overline{u}}\xi_x + \overline{\overline{v}}\xi_y + \overline{\overline{a}}\left(\xi_x^2 + \xi_y^2\right)^{\frac{1}{2}} \\ \overline{\overline{u}}\xi_x + \overline{\overline{v}}\xi_y - \overline{\overline{a}}\left(\xi_x^2 + \xi_y^2\right)^{\frac{1}{2}} \end{bmatrix}. \tag{6.17}$$

The right eigenvectors are

$$\overline{\overline{\boldsymbol{r}}}_1 = \begin{bmatrix} 1 \\ \overline{\overline{u}} \\ \overline{\overline{v}} \\ \dfrac{\left(\overline{\overline{u^2}} + \overline{\overline{v^2}}\right)}{2} \end{bmatrix}, \qquad \overline{\overline{\boldsymbol{r}}}_2 = \begin{bmatrix} 0 \\ \overline{\overline{\rho}}\hat{\xi}_y \\ -\overline{\overline{\rho}}\hat{\xi}_x \\ \overline{\overline{\rho}}\left(\overline{\overline{u}}\hat{\xi}_x - \overline{\overline{v}}\hat{\xi}_y\right) \end{bmatrix},$$

$$\overline{\overline{\boldsymbol{r}}}_3 = \frac{\overline{\overline{\rho}}}{2\overline{\overline{a}}} \begin{bmatrix} 1 \\ \overline{\overline{u}} + \overline{\overline{a}}\hat{\xi}_x \\ \overline{\overline{v}} + \overline{\overline{a}}\hat{\xi}_y \\ \overline{\overline{h_t}} + \overline{\overline{a}}\left(\overline{\overline{u}}\hat{\xi}_x - \overline{\overline{v}}\hat{\xi}_y\right) \end{bmatrix}, \qquad \overline{\overline{\boldsymbol{r}}}_4 = \frac{\overline{\overline{\rho}}}{2\overline{\overline{a}}} \begin{bmatrix} 1 \\ \overline{\overline{u}} - \overline{\overline{a}}\hat{\xi}_x \\ \overline{\overline{v}} - \overline{\overline{a}}\hat{\xi}_y \\ \overline{\overline{h_t}} - \overline{\overline{a}}\left(\overline{\overline{u}}\hat{\xi}_x + \overline{\overline{v}}\hat{\xi}_y\right) \end{bmatrix}. \tag{6.18}$$

The wave amplitude δw is

$$\delta w_1 = 2\left(\delta u_1 + \frac{\delta u_1 \overline{\overline{h}}_t - \delta u_2 \overline{\overline{u}} - \delta u_3 \overline{\overline{v}} + \delta u_4}{-2\overline{\overline{h}}_t + \overline{\overline{u^2}} + \overline{\overline{v^2}}}\right)$$

$$\delta w_2 = \frac{\delta u_1\left(\overline{\overline{v}}\hat{\xi}_x - \overline{\overline{u}}\hat{\xi}_y\right) + \delta u_2\hat{\xi}_y - \delta u_3\hat{\xi}_x}{\overline{\overline{\rho}}}$$

$$\delta w_3 = \left(\delta u_2\hat{\xi}_x + \delta u_3\hat{\xi}_y - \delta u_1\left(\overline{\overline{u}}\hat{\xi}_x + \overline{\overline{v}}\hat{\xi}_y\right) - \overline{\overline{a}}\left(\frac{\delta u_1\left(\overline{\overline{u^2}} + \overline{\overline{v^2}}\right) - 2\delta u_2\overline{\overline{u}} - 2\delta u_3\overline{\overline{v}} + 2\delta u_4}{-2\overline{\overline{h_t}} + \overline{\overline{u^2}} + \overline{\overline{v^2}}}\right)\right)/\overline{\overline{\rho}} \tag{6.19}$$

$$\delta w_4 = -\left(\delta u_2\hat{\xi}_x + \delta u_3\hat{\xi}_y - \delta u_1\left(\overline{\overline{u}}\hat{\xi}_x + \overline{\overline{v}}\hat{\xi}_y\right) + \overline{\overline{a}}\left(\frac{\delta u_1\left(\overline{\overline{u^2}} + \overline{\overline{v^2}}\right) - 2\delta u_2\overline{\overline{u}} - 2\delta u_3\overline{\overline{v}} + 2\delta u_4}{-2\overline{\overline{h_t}} + \overline{\overline{u^2}} + \overline{\overline{v^2}}}\right)\right)/\overline{\overline{\rho}}$$

where

$$\delta \boldsymbol{u} = \boldsymbol{U}^R_{i+\frac{1}{2}} - \boldsymbol{U}^L_{i+\frac{1}{2}} \tag{6.20}$$

and $\boldsymbol{U}$ is the vector of conserved variables in physical space. The quantities with double bars are Roe averaged variables and are calculated using

$$\overline{\overline{\rho}} = R_{i+\frac{1}{2}}\rho^L \tag{6.21}$$

$$\overline{\overline{u}} = (R_{i+\frac{1}{2}}u^R + u^L)/(R_{i+\frac{1}{2}} + 1) \tag{6.22}$$

$$\overline{\overline{v}} = (R_{i+\frac{1}{2}}v^R + u^L)/(R_{i+\frac{1}{2}} + 1) \tag{6.23}$$

$$\overline{\overline{h_t}} = (R_{i+\frac{1}{2}}h_t^R + h_t^L)/(R_{i+\frac{1}{2}} + 1) \tag{6.24}$$

$$\overline{\overline{a}} = \sqrt{(\gamma - 1)\left(\overline{\overline{h_t}} - \frac{1}{2}\left(\overline{\overline{u^2}} + \overline{\overline{v^2}}\right)\right)} \tag{6.25}$$

where $R_{i+\frac{1}{2}} = \sqrt{\rho^R/\rho^L}$. The right and left state vectors are the solution vectors on either size of the flux interface and is calculated using MUSCL extrapolation.

6.1.3 MUSCL extrapolation

To increase the order of accuracy of Roe's upwind scheme, a larger stencil is used to extrapolate data to the cell faces. The higher-order extension is often referred to as MUSCL extrapolation and was first introduced by Van Leer [36]. The left state is calculated by

$$U^L_{i+\frac{1}{2}} = U_i + \frac{\epsilon}{4}\big((1-\kappa)(U_i - U_{i-1}) + (1+\kappa)(U_{i+1} - U_i)\big), \tag{6.26}$$

and the right state is calculated by

$$U^R_{i-\frac{1}{2}} = u_{i+1} - \frac{\epsilon}{4}\big((1-\kappa)(U_{i+2} - U_{i+1}) + (1+\kappa)(U_{i+1} - U_i)\big). \tag{6.27}$$

The two parameters ϵ and κ specify the order of accuracy and the stencil used, see Table 6.1 and Table 6.2. All Euler solutions included in this study are second-order accurate and fully upwinded, $\epsilon = 1$ and $\kappa = -1$.

Table 6.1. Options for MUSCL extrapolation parameter ϵ

ϵ	Order of Accuracy
0	First-order
1	Second-order or higher

Table 6.2. Options for MUSCL extrapolation parameter κ

κ	Order of Accuracy	Stencil
-1	Second-order	Fully upwind
0	Second-order	Upwind biased
1/2	Second-order	QUICK scheme [37]
1/3	Third-order	Upwind biased
1	Second-order	Average state

6.1.4 Boundary Conditions

Three different boundary conditions are implemented in the 2D Euler solver required for the exact solutions given in Chapter 5: Dirichlet, slip wall, and outflow. Dirichlet boundary conditions specify the exact flow quantities at the face center so that the exact flux can be calculated. The slip wall boundary condition specifies zero flux through the cell boundary by setting the contravarient velocity perpendicular to the boundary to zero. For example, the flux calculated at a wall along the $\xi = 1$ boundary is calculated by

$$\boldsymbol{f}_{1/2}^{L} = \frac{1}{J}\begin{bmatrix} 0 \\ p_{1/2}\xi_x \\ p_{1/2}\xi_y \\ 0 \end{bmatrix}. \tag{6.28}$$

The pressure is calculated at the face using a second-order extrapolation

$$p_{1/2} = \frac{3}{2}p_1 - \frac{1}{2}p_2. \tag{6.29}$$

For the outflow boundary condition, pressure is specified. Depending on the Mach number of the flow, the pressure may or may not be used. If the flow is supersonic, the eigenvalues are all positive and all flow information is propagated downstream and the pressure is not used. If the flow is subsonic, one eigenvalue is negative and pressure specified at the boundary is propagated upstream.

6.1.5 Code Verification

To verify that no coding mistakes are present, the most rigorous code verification procedure is the order of accuracy test and is applied to all options available in the 2D Euler code. The order of accuracy test verifies that the numerical solution approaches the exact solution to the governing equations at the rate specified by the discretization scheme. The exact solution is required to calculate the observed order of accuracy which should approach the formal order of accuracy as the mesh is refined. When the exact solution to the PDEs is known, the observed order of accuracy is calculated by

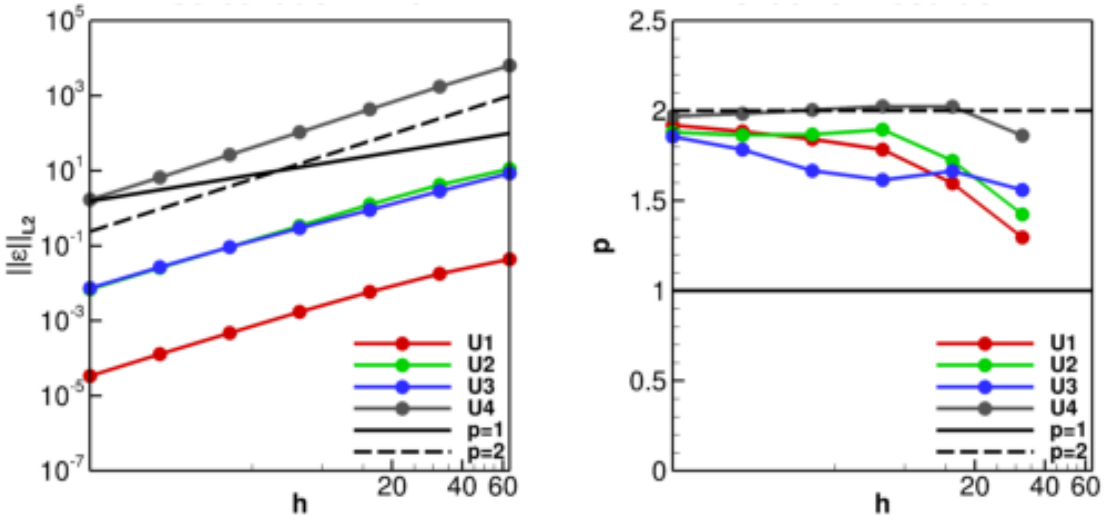
$$\hat{p} = \frac{ln\left(\frac{\left|\left|\varepsilon_{rh}\right|\right|_{L_2}}{\left|\left|\varepsilon_{h}\right|\right|_{L_2}}\right)}{ln(r)} \tag{6.30}$$

where ε_h and ε_{rh} is the exact discretization error for the fine and coarse mesh, respectively.

Three of the exact solutions given in Chapter 5 are used for code verification. These are the supersonic manufactured solution, the subsonic manufactured solution, and supersonic vortex flow. Ringleb's flow is excluded because the boundary conditions are identical to supersonic vortex flow.

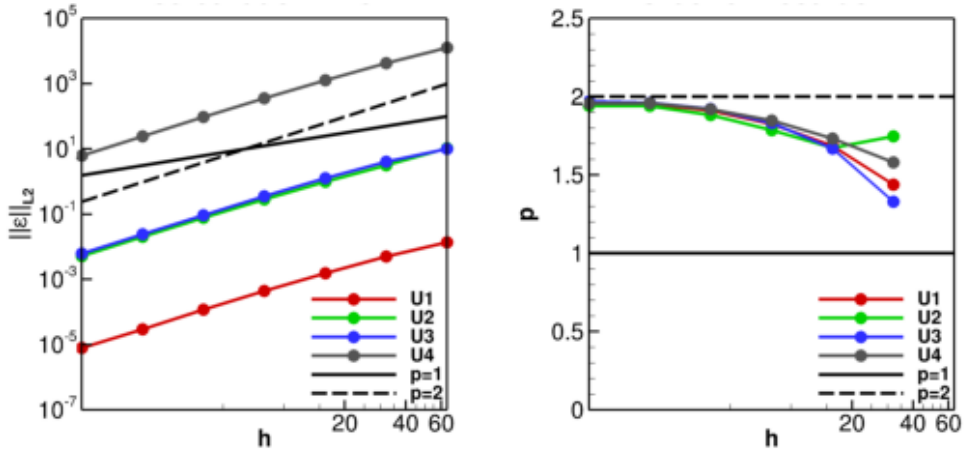
6.1.5.1 Manufactured Solutions

The manufactured solutions have Dirichlet boundary conditions so code verification tests the primary routines and discretization scheme. The results for the subsonic manufactured solution and the supersonic manufactured solutions are shown in Figure 6.1 and Figure 6.2, respectively. The observed order of accuracy for both cases clearly approaches second-order accurate. The subsonic manufactured solution has negative velocities in the entire domain, so the results shown in Figure 6.1 confirm that only half of the flux calculations are correct. A similar subsonic case with all positive velocities was used to confirm that other half of the subsonic flux calculations are correct. The results which are not shown had almost identical errors and was second-order accurate. The supersonic manufactured solution has a positive velocity in the entire domain verifying only half of the supersonic flux terms. A supersonic manufactured solution with negative velocity was used to verify the other half of the flux calculation. The supersonic manufactured solution results with negative velocities had almost identical discretization errors and was second-order accurate. The results for the manufactured solutions verify the basic discretization scheme and Dirichlet boundary conditions.



a) Discretization Error **b) Order of Accuracy**

Figure 6.1. Discretization error and observed order of accuracy for the subsonic manufactured solution

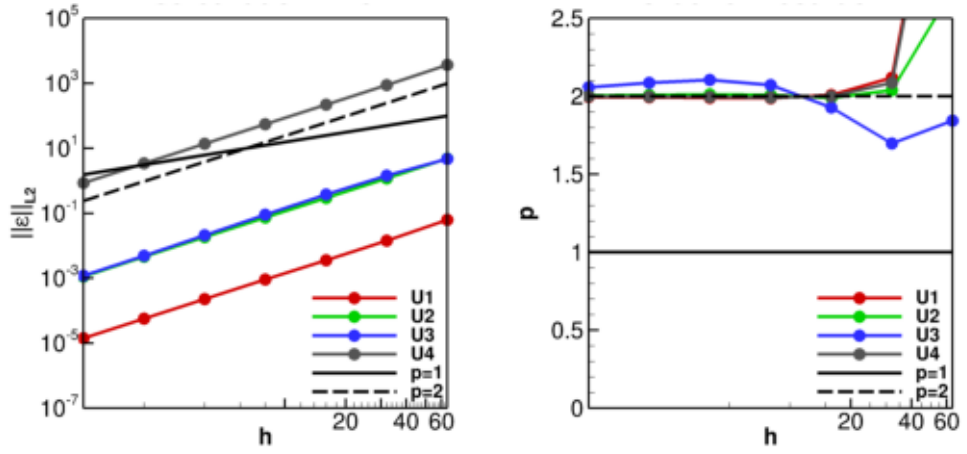


a) Discretization Error **b) Order of Accuracy**

Figure 6.2. Discretization error and observed order of accuracy for the supersonic manufactured solution

6.1.5.2 Supersonic Vortex Flow

Supersonic vortex flow uses Dirichlet, wall, and outflow boundary conditions. The results are shown in Figure 6.3. The order of accuracy clearly approaches second-order, which partially verifies the wall and outflow boundary conditions. The wall boundary conditions are along two of the boundaries, and the outflow is along one of the boundaries. To verify that the boundary conditions are properly implemented for other boundaries, the order of the grid nodes are modified. This changes the mapping from physical to computational space so that the boundary conditions are applied to different computational boundaries even though the physical mesh remains the same. Three additional cases with modified grid nodes are run to fully verify all of the boundary conditions. The resulting discretization error and observed order of accuracy were nearly identical to the results shown in Figure 6.3.



a) Discretization Error **b) Order of Accuracy**

Figure 6.3. Discretization error and observed order of accuracy for supersonic vortex flow

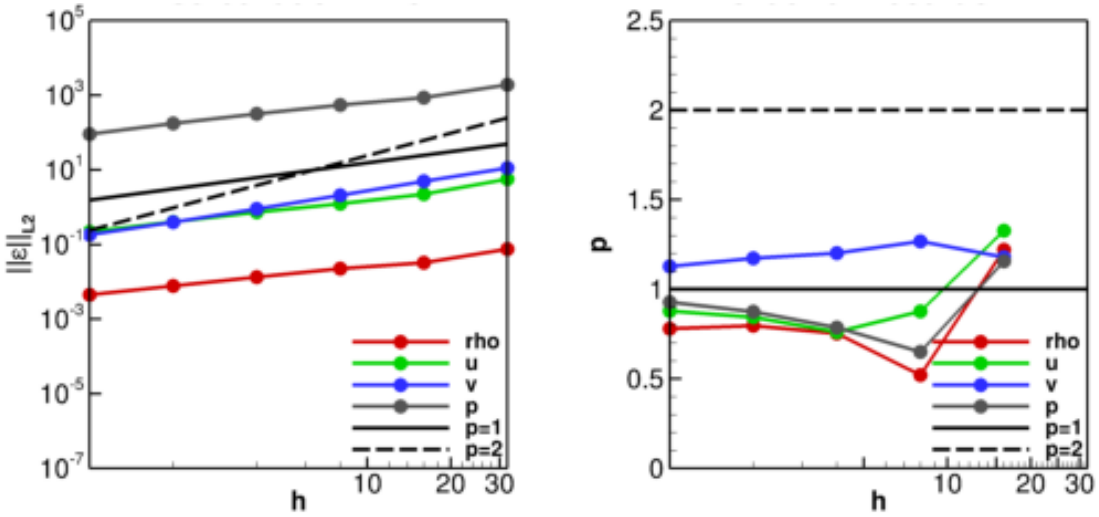
6.2 Code Verification of FLUENT

The flow solver used to solve the Navier-Stokes equations is ANSYS/FLUENT [33]. There is no known code verification of FLUENT so some limited code verification is done using two different cases. The first case is supersonic vortex flow. This case will test the Euler equations using Roe's discretization scheme. A second case is a turbulent flat plate solved with the *k-ω SST* turbulence model. This case is similar to the numerical benchmark case described in Chapter 7.

6.2.1 Supersonic Vortex Flow

Supersonic vortex flow is chosen as a verification case because the flow is perpendicular to the inflow making it relatively easy to specify the inflow velocity profile. The velocity profile is specified using a User Defined Function (UDF) which is an external file written in C and interpreted or compiled by FLUENT. The UDF used to specify the inflow and the exact case setup in FLUENT is in Appendix 2.

The results returned from FLUENT are cell centered data and are interpolated to node centered data using "inverse-distance" interpolation available in Tecplot 360 [38]. The boundary nodes are excluded because it is an extrapolation instead of an interpolation and results in significantly higher error associated with this post processing step. The L_2-norm of discretization error is calculated and results are shown in Figure 6.4. The order of accuracy for this case approaches one for all variables. The numerical solution does approach the exact solution; however, at a reduced rate providing minimal confidence in the numerical results.



a) Discretization Error **b) Order of Accuracy**

Figure 6.4. Discretization error and observed order of accuracy for supersonic vortex flow solved using FLUENT

6.2.2 Turbulent Flat Plate

A second code verification method which is used to provide some confidence in the numerical solution is code-to-code comparison. The specific case is a turbulent flat plate solved using the *k-ω SST* turbulence model and is solved using FUN3D [40] and CFL3D [41] and was published explicitly for code verification purposes [39]. Rumsey and Thomas [42**]** applied manufactured solutions to FUN3D and CFL3D using the Spalart-

Allmaras turbulence model showing second-order accuracy. The *k-ω SST* turbulence model has not been verified however. Code-to-code comparison is not as rigorous as the order of accuracy test but can be used to provide some confidence in the numerical results from FLUENT.

The turbulent flat plate is solved on a rectangular domain with a flat plate length of 2 feet (0.6 meters) and a height of 1 foot (0.3 meters). The Reynolds number at 0.3 meters is 5×10^6. The details of the case setup are given in Figure 6.5. The turbulent farfield boundary conditions are

$$k_{farfield} = 9 \times 10^{-9} a_\infty^2 \tag{6.31}$$

$$\omega_{farfield} = 1 \times 10^{-6} \frac{\rho_\infty \infty a_\infty^2}{\mu_\infty} \tag{6.32}$$

All grids are given in Ref. 39. Specific variables are available for comparison. The first is the coefficient of friction at $x = 0.295$ meters and the second is the coefficient of drag on the flat plate. The results are shown in Figure 6.6. The results from FLUENT asymptotically approach similar results as those from FUN3D and CFL3D providing some confidence in the numerical results from FLUENT.

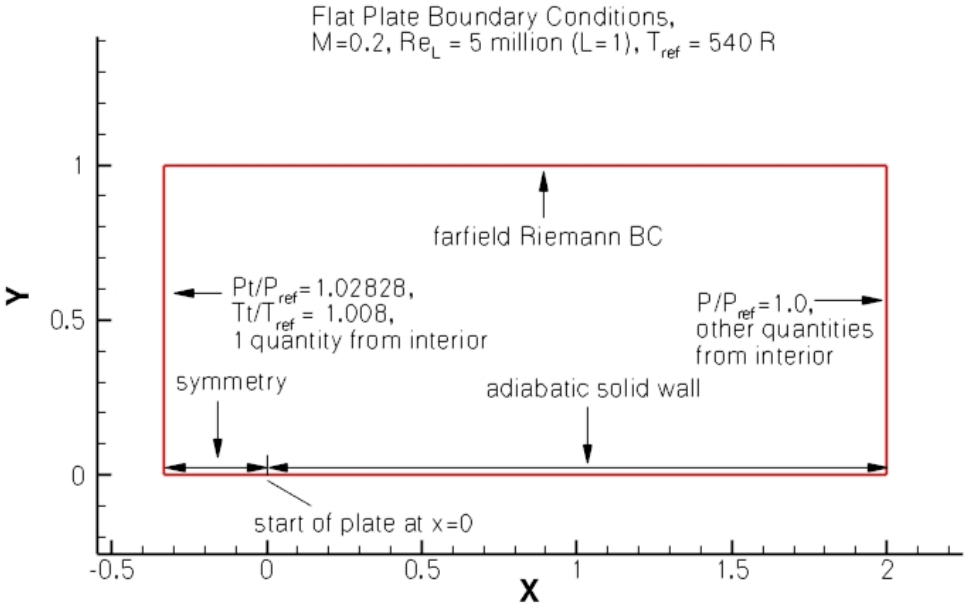
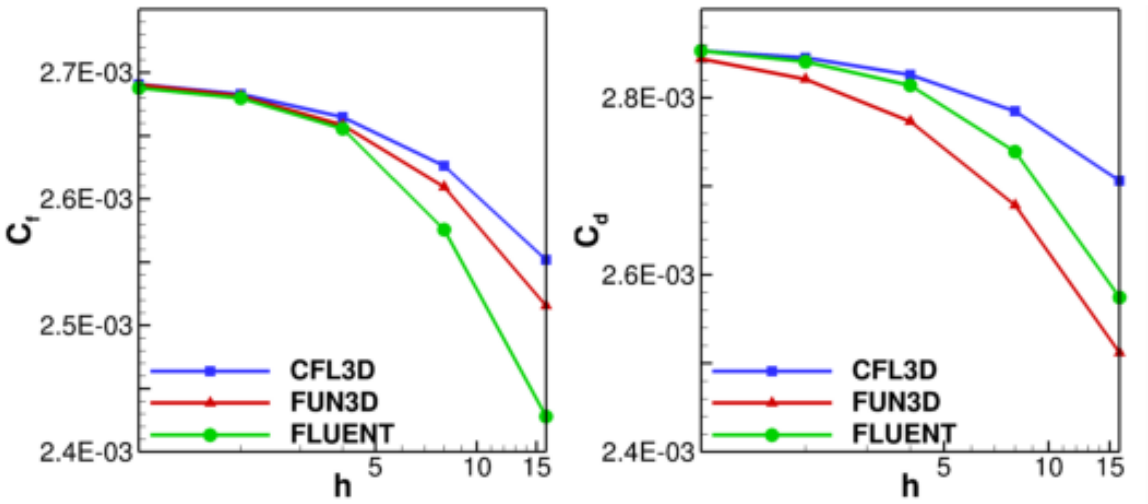
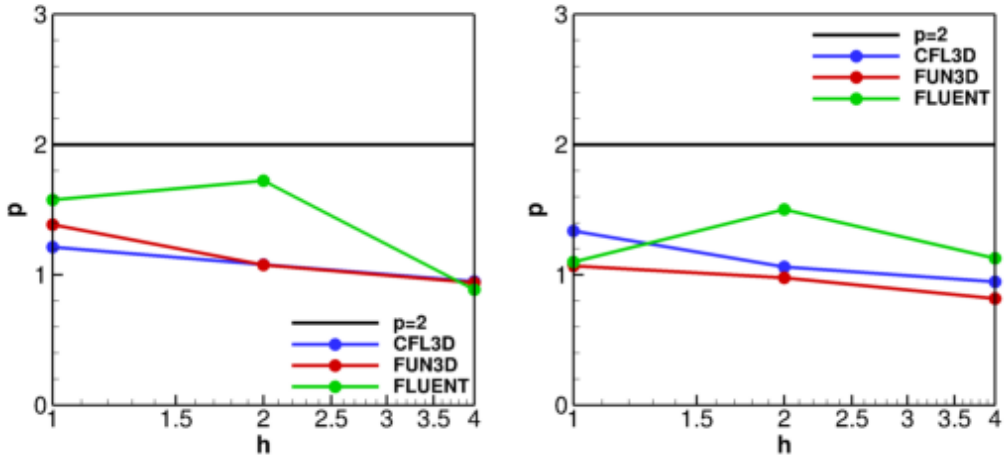


Figure 6.5 Turbulent flat plate case setup [39]



a) Coefficient of Friction (x=0.295m) **b) Coefficient of Drag**

Figure 6.6. Convergence of flow variables for the turbulent flat plate



a) Coefficient of Friction (x=0.295m) **b) Coefficient of Drag**

Figure 6.7. Observed order of accuracy for the turbulent flat plate

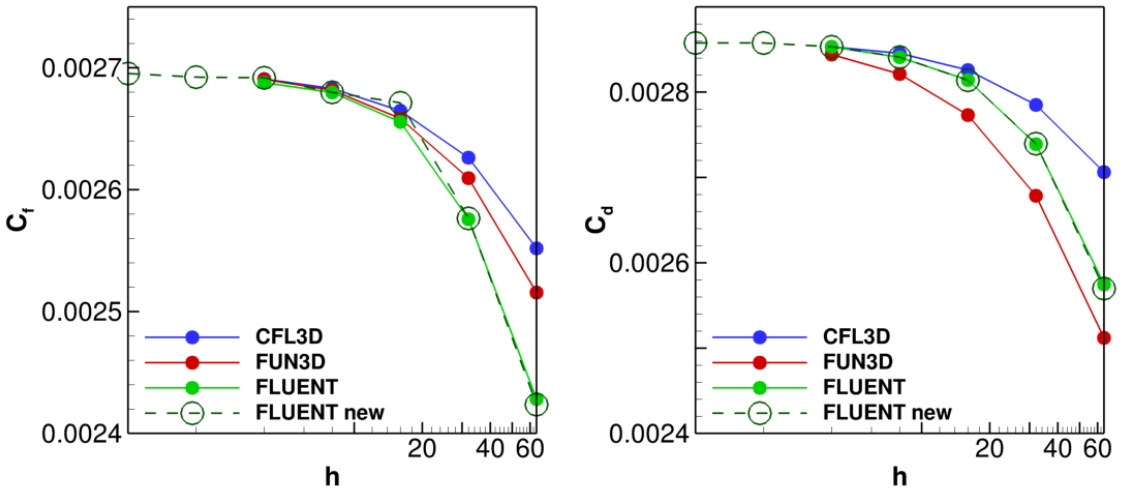
7 Numerical Benchmarks

To evaluate discretization error estimators, the exact solution to the differential equations must be known. Exact solutions, however, exists for relatively simple problems. To evaluate error estimators for more complicated flows a numerical benchmark must be created. A numerical benchmark must have numerical errors that are significantly smaller than the numerical errors in the solutions that are used to evaluate the error and uncertainty estimators. Extending the rule-of-thumb given in Roy [4], the total numerical errors present in the benchmark should be at least two orders of magnitude lower than the numerical errors in the solutions that are being assessed. This means that the benchmark numerical solution must be computed on an extremely fine mesh, with double precision computations (or higher), and with low iterative errors. Furthermore, in order to establish that the numerical benchmark solution is in the asymptotic convergence range, iterative and round-off errors must also be negligible compared to the discretization error in the benchmark solution. This places further requirements on iterative convergence and round-off error in the benchmark solution. A numerical benchmark is created for a turbulent flat plate and a laminar flat plate.

7.1 Turbulent Flat Plate

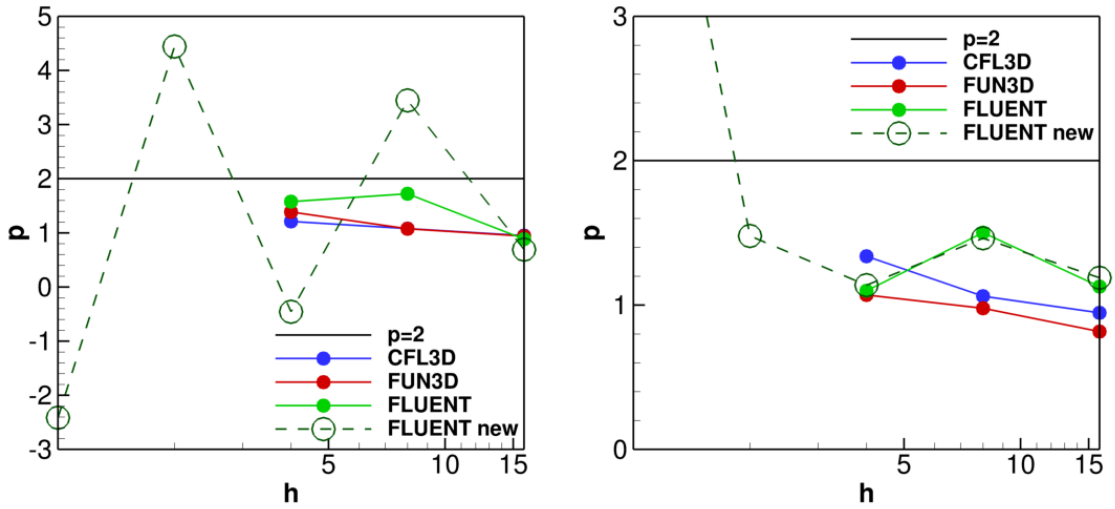
The starting point for the turbulent flat plate benchmark is the turbulent flat plate case use for the code verification of FLUENT discussed in Section 6.2.2. Several grid levels were used for code verification which are also used to evaluate the discretization error and uncertainty estimators. As a starting point, the benchmark solution should be solved on a mesh with numerical errors two orders of magnitude lower than the discretization error on the finest grid (545x385) used for evaluation. For a second-order accurate discretization method, the refinement factor required is about eight. For an asymptotic solution, this refinement factor will result in a decrease in discretization error by about a factor of 64 resulting in the finest mesh of 4353x3073 which has a total of 13.4 million grid nodes.

The original 545x385 grid was initially refined by a factor of eight using interpolation; however, this resulted in an unsatisfactory grid due to interpolation error. Instead, the numerical benchmark grid was created from scratch. The new grid was created so that a refinement by a factor of eight will result in as close of a match as possible to the original 545x385 grid. The results are compared to the previous results in Figure 7.1 and Figure 7.2 where the finest mesh is 2177x1537 (*h=1* in the referenced figures). The convergence for the coefficient of drag matches the previous FLUENT results almost exactly. The corresponding order of accuracy also matches well. The convergence for coefficient of friction matches reasonably well with one obvious mismatched data point at $h = 16$. The slightly different results are not unexpected since coefficient of friction is a local quantity and will be more sensitive to small changes in the mesh compared to the coefficient of drag. The observed order of accuracy for both quantities is not asymptotic and is highly oscillatory.



a) Coefficient of Friction (x=0.295m) **b) Coefficient of Drag**

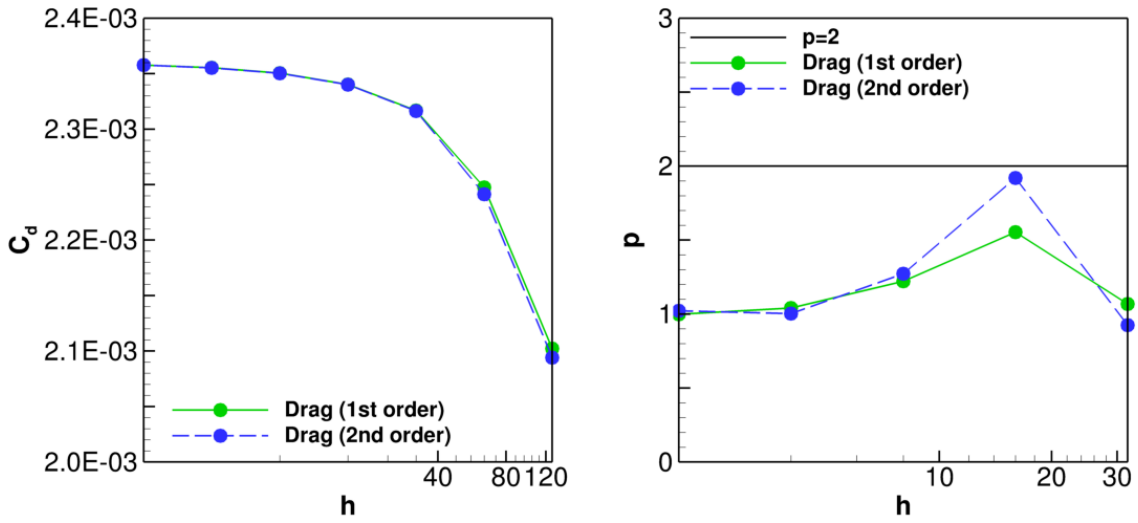
Figure 7.1. Convergence of flow variables for the turbulent flat plate compared to the solutions on the new flat plate grid



a) Coefficient of Friction (x=0.295m) **b) Coefficient of Drag**

Figure 7.2. Order of accuracy for flow variables for the turbulent flat plate compared to the orders of accuracy on the new flat plate grid

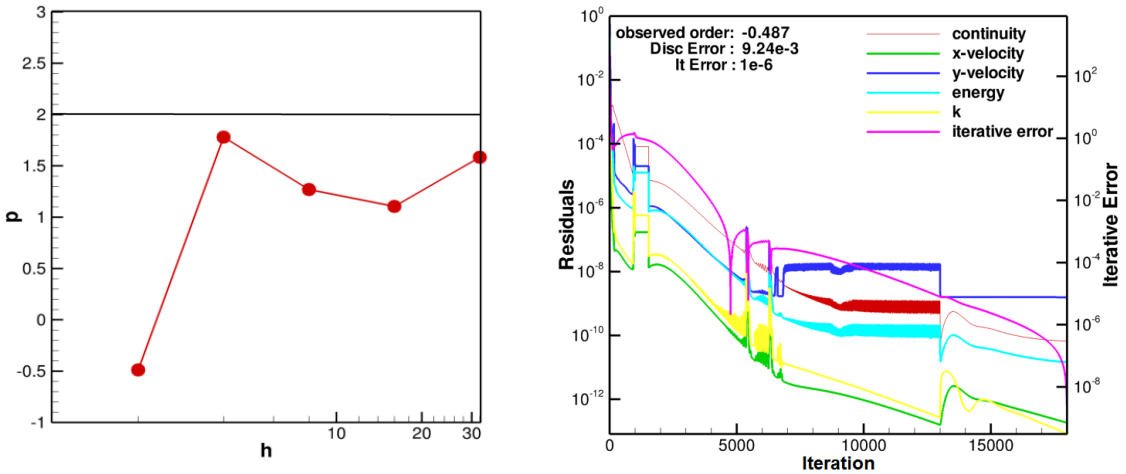
The lack of asymptotic behavior is attributed to the leading edge singularity where the plate transitions from a slip wall to a no-slip wall. To improve the solution, the slip wall region of the mesh was removed with all other parameters remaining constant. The results for the truncated flat plate are shown in Figure 7.3. Both a first-order accurate solution and second-order accurate solution were computed. The results show that the solution is clearly asymptotic; however, the order of accuracy approaches first-order. The non-linear leading edge singularity is due to the sudden change in the y-velocity component and reduces the formal order of accuracy to first-order [43]. Results were similar for the local coefficient of friction.



a) Coefficient of Drag **b) Observed Order of Accuracy**

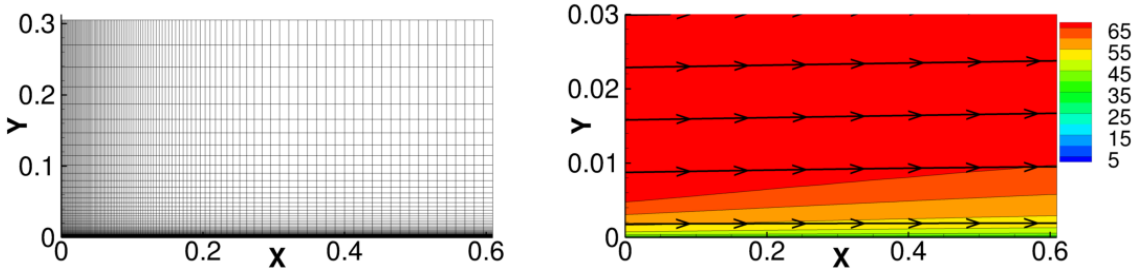
Figure 7.3. Coefficient of drag and observed order of accuracy for truncated flat plate

To remove the leading edge singularity, a boundary layer profile is specified at the inlet. This profile is extracted from the numerical solution for the non-truncated flat plate solution solved on the finest mesh 4353x3073 at a Reynolds number of five million (the grid node closest to x=0.3 meters). The results are shown in Figure 7.4. The order of accuracy for coefficient of drag shows asymptotic behavior up to the 1793x1537 grid (h=2) then the order of accuracy suddenly drops. The leading edge singularity is no longer present and the iterative error shown in Figure 7.4b is almost four orders of magnitude lower than the estimated discretization error. The exact reason for the sudden decrease in discretization error is now known. Since the solution is showing asymptotic behavior 897x769 (*h=4*) grid is used as the numerical benchmark and the error estimators are evaluated using the 29x25, 57x49, and 113x97 grid levels. The final benchmark solution is shown in Figure 7.5.



a) Order of Accuracy **b) Convergence history**

Figure 7.4. Order of accuracy for coefficient of drag and iterative convergence history



a) 137x97 mesh **b) x-velocity (m/s)**

Figure 7.5. Truncated mesh for the flat plate and the solution on the 113x97 node mesh with x-velocity contours and streamlines (Re_x=5e6 at x=0 m)

7.2 Laminar Flat Plate

A similar approach is taken to create a numerical benchmark for a laminar flat plate. Blasius's zero pressure gradient, boundary layer solution is used to define the inflow conditions but cannot be used as an exact solution because Blasius's solutions solves the boundary layer equations and not the full Navier-Stokes equations. The truncated flat plate grids used for the turbulent benchmark are also used for the laminar flat plate. The edge velocity is 0.10 m/s, the density is 1.1614402 kg/m^3, and the kinetic viscosity is 0.15524232e-5 $Pa-s$. The Reynolds number of the inflow boundary condition is determined based on the specified ratio of domain height to maximum boundary layer height of 20 to reduce the effect of the upper boundary on the solution. The inflow Reynolds number is 6393 where

$$Re_x = \frac{\rho U_e x}{\mu}. \tag{7.1}$$

The inflow profile is calculated from Blasius's solution defined by the differential equation

$$f''' + ff'' = 0. \tag{7.2}$$

where f is a function of the non-dimensional coordinate η

$$\eta = y\sqrt{\frac{U_e}{2\nu x}}. \tag{7.3}$$

The stream function is defined by

$$\psi = \sqrt{2\nu U_e x} f(\eta) \tag{7.4}$$

and the velocity components are calculated by

$$u = U_e f'(\eta) \tag{7.5}$$

$$v = \sqrt{\frac{\nu U_e}{2x}}\left(\eta f'(\eta) - f(\eta)\right). \tag{7.6}$$

The ordinary differential equation is solved using the method defined in Ref. 1. This method breaks the third-order differential equation into a system of three first-order differential equations using $g_1 = f$, $g_2 = f'$, and $g_3 = f''$ as

$$g_1' = g_2 \tag{7.7}$$

$$g_2' = g_3 \tag{7.8}$$

$$g_3' = -g_1 g_3. \tag{7.9}$$

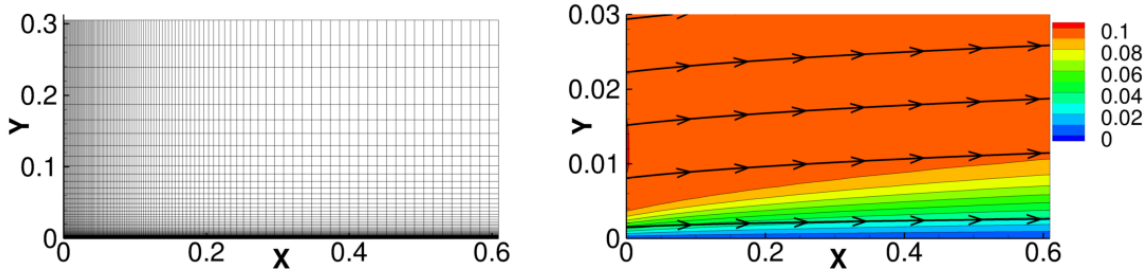
The boundary conditions are

$$g_1(0) = 0 \tag{7.10}$$

$$g_2(0) = 0 \tag{7.11}$$

$$\lim_{\eta \to \infty} g_2(\eta) = 1.0. \tag{7.12}$$

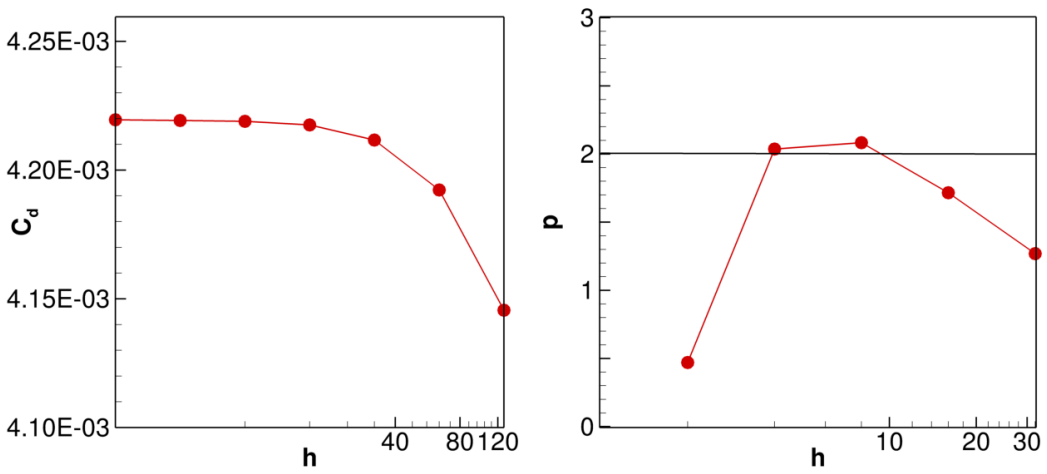
This system of equations is solved using a fourth-order Runge-Kutta integration scheme. A sample mesh and numeric solution are shown in Figure 7.6.



a) 137x97 mesh **b) x-velocity (m/s)**

Figure 7.6. Truncated mesh for the flat plate and the solution on the 113x97 node mesh with x-velocity contours and streamlines (Re_x=6393 at x=0 m)

The resulting coefficient of drag for the laminar flat plate case is shown in Figure 7.7. The observed order of accuracy asymptotically approaches the formal order of two until the 1793x1537 (*h=2*) grid level. The observed order of accuracy suddenly decreases to approximately 0.5. Iterative error shown in Figure 7.8 is on the order of 1e-17 and can be considered negligible. The cause of the sudden decrease is unknown; however, the behavior is very similar to that seen for the turbulent flat plate. Since asymptotic behavior is apparent up to the 897x769 grid, this is the numeric benchmark and the grid levels used to evaluate the error and uncertainty estimators are 29x25, 57x49, and 113x97.



a) Coefficient of drag **b) Observed order of accuracy**

Figure 7.7. Results for the laminar flat plate showing the coefficient of drag and the observed order of accuracy

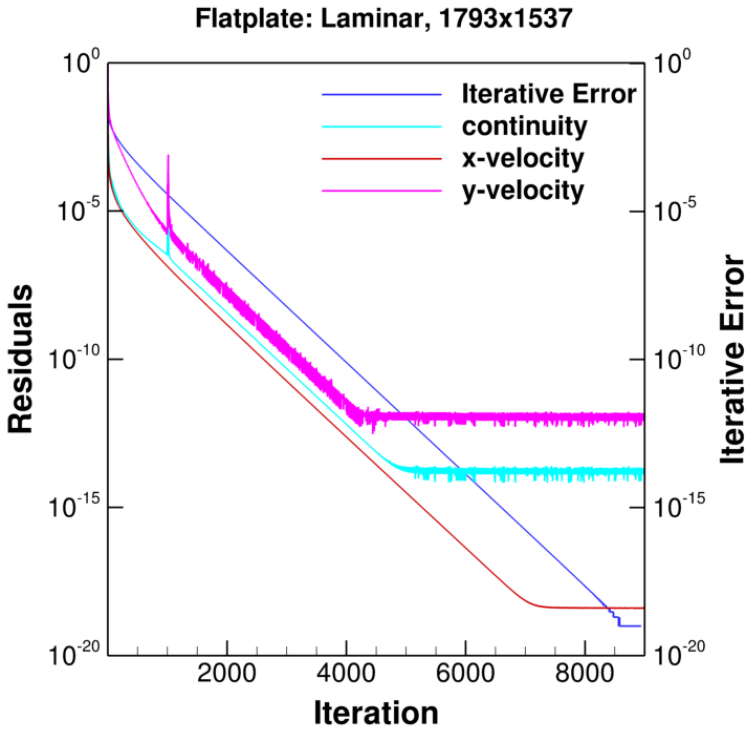


Figure 7.8. Iterative error in coefficient of drag for the h=2 (1793x1537) laminar flat plate

8 Error and Uncertainty Metrics

The goal of uncertainty estimation is to account for the additional uncertainty that is associated with non-asymptotic solutions and should bracket the exact solution. If an estimate brackets the exact solution to the governing equations, the estimate is conservative. An uncertainty estimate should be conservative for 95 percent of the data. The 95 percent conservativeness could be achieved by choosing a large factor of safety such as 1000; however, this choice of factor of safety does not provide useful information about the solution. The effectivity index is used to measure the accuracy of the estimate. This metric is applied to both the error estimate and the uncertainty estimate.

8.1 Conservativeness

The conservativeness of an uncertainty estimate is the primary metric of interest and is the fraction of the data that is bracketed by the estimate where a value of one means that all data was bracketed. The uncertainty estimators should be 95 percent conservative regardless of the asymptotic nature of the solution; however, since the error estimate becomes more accurate as the mesh is refined, the conservativeness should tend to approach one. An example of conservativeness versus mesh spacing is shown in Figure 8.1.

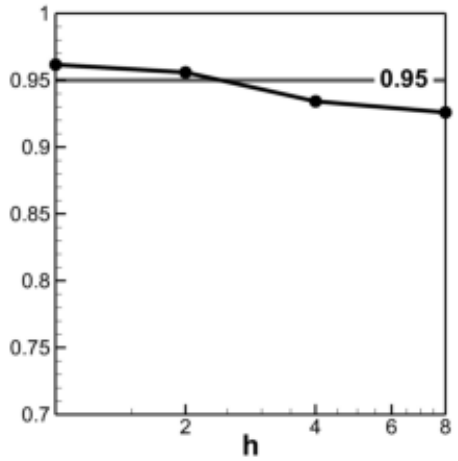


Figure 8.1. Conservativeness versus mesh spacing

8.2 Effectivity Index

8.2.1 Error Effectivity Index

The effectivity index was first used for finite element methods [44] and is calculated by

$$\theta_{\Omega} = \frac{|\bar{\varepsilon}_h|}{|\varepsilon_h|} \tag{8.1}$$

where $\bar{\varepsilon}_h$ is the discretization error estimate and ε_h is the exact discretization error. For local error estimation, the effectivity index is the L_2-norm of the error estimate divided by the L_2-norm of the exact error

$$\theta_{L_2} = \frac{\|\bar{\varepsilon}_h\|_{L_2}}{\|\varepsilon_h\|_{L_2}}. \tag{8.2}$$

As a mesh is refined and the solution becomes more asymptotic, the effectivity index approaches one. The effectivity index can vary from near one to a couple orders of magnitude greater than one depending on the asymptotic nature of the solution and the error estimator. To compare the effectivity index for various solutions and error estimators, the inverse of the effectivity index is plotted. See Figure 8.2a for an example. In this example the inverse effectivity index is approaching one from below which means that the error is underestimated. If the effectivity index approaches one from above the error is overestimated.

8.2.2 Uncertainty Effectivity Index

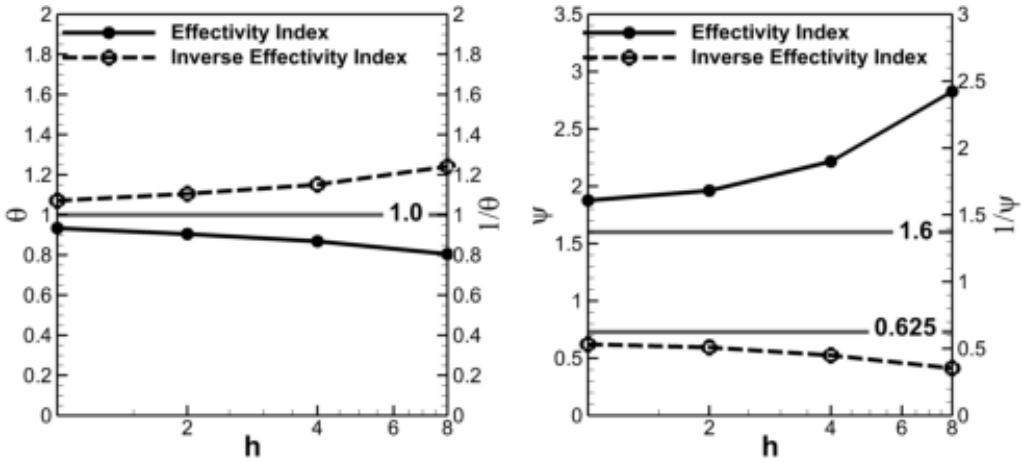
An equivalent effectivity index is used for uncertainty estimation and is defined as

$$\psi_{\Omega} = \frac{|U|}{|\varepsilon_h|}. \tag{8.3}$$

The uncertainty effectivity index for local estimates are calculated using the L_2-norm

$$\psi_{L_2} = \frac{\|U\|_{L_2}}{\|\varepsilon_h\|_{L_2}}. \tag{8.4}$$

As the solution becomes more asymptotic, the uncertainty effectivity index approaches the factor of safety used for the particular uncertainty estimate. For convenience, the inverse of the uncertainty effectivity index is plotted. Since the asymptotic behavior depends on the factor of safety used in the estimate, the inverse of the uncertainty effectivity index depends on the inverse of the factor of safety. See Figure 8.2b as an example for the supersonic manufactured solution density variable.



a) Error Effectivity Index **b) Uncertainty Effectivity Index**

Figure 8.2. Example of the effectivity index for error and uncertainty estimation

8.3 Probability Density Function

The effectivity index is a single global metric calculated from local data with no information provided about the distribution of the local data. Probability Density Functions (PDFs) are created for each of the error and uncertainty estimators to show the distribution of the (inverse) local error effectivity index. The local effectivity index at each grid node is calculated using

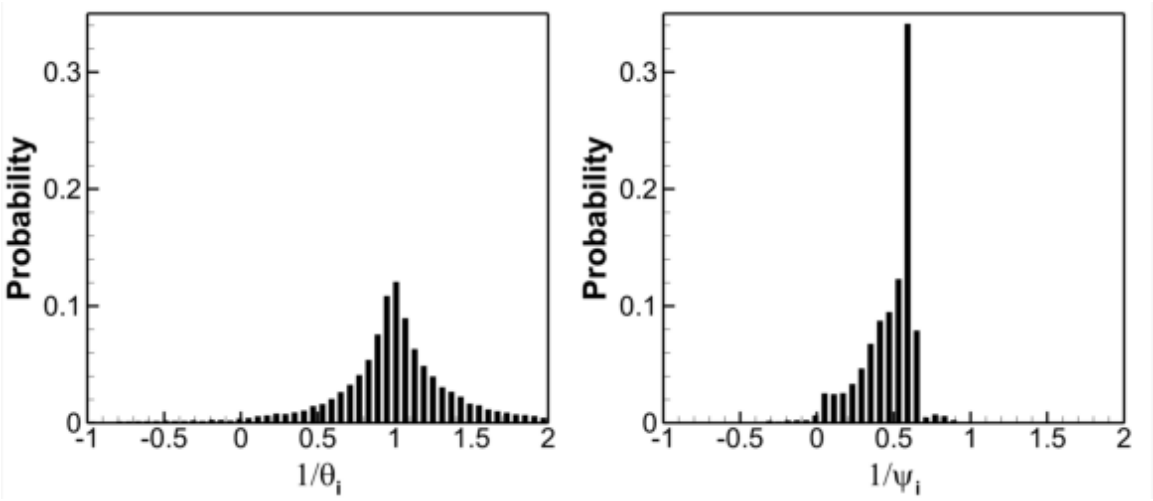
$$\theta_i = \frac{\overline{\varepsilon}_h}{\varepsilon_h}. \tag{8.5}$$

To correspond to the representation of the globally calculated effectivity index, the inverse of the local effectivity index is used to create the PDF where a value greater than one means an underestimate of the error. The data calculated from Equation (8.5) is limited to values between -1 and 2 and is divided into a total of 50 equally-spaced bins. The number of data points with an effectivity index within each bin is counted. All data is then normalized by the total number of data points.

An equivalent PDF is also created for the uncertainty effectivity index where the sign of the error is also taken into consideration.

$$\psi_i = sign(1, u_{rh} - u_h)\frac{U}{\varepsilon_h} \tag{8.6}$$

An example of an error PDF and uncertainty PDF is given in Figure 8.3a and Figure 8.3b, respectively. The data chosen as an example corresponds to the $h = 1$ data in Figure 8.2.



a) Error PDF **a) b) Uncertainty PDF**

Figure 8.3. Example of PDFs created to investigate the local distribution of the effectivity index

8.4 Data Visualization

An important aspect of this study is to investigate the asymptotic behavior of the uncertainty estimators which requires that various data from several cases are combined in such a way that the asymptotic behavior is captured. Various metrics were considered

including the average order of accuracy, the number of mesh nodes, percentage of monotonically converging nodes, and the average distance from the formal order of accuracy, see Figure 8.4.

The average order of accuracy failed to capture the asymptotic range because very large orders of accuracy canceled the negative orders of accuracy resulting in an average near the formal order of accuracy. A global order of accuracy similar to that used by Cadafalch *et al.* [19] and described in Section 3.2.3 was also considered. This method also failed because large formal orders of accuracy are not penalized for non-asymptotic behavior because of the imposed formal order of accuracy limit in Equation (3.16).

The number of mesh nodes failed to capture the asymptotic behavior because there is no relationship between the solution and number of mesh nodes. The example shown in Figure 8.4 is for a single case and shows the distribution of data decreasing as the number of mesh nodes is increased; however, other simulations with different flow features will not be asymptotic with only 4000 mesh nodes.

The last two metrics considered were able to capture the asymptotic behavior of the solution. These two quantities are compared against each other in Figure 8.5 showing a clear relationship between the two. This suggests that either could be a suitable metric to represent the asymptotic convergence of numerical solutions; however, the average distance from the asymptotic range is chosen because of a few characteristics seen as an advantage over using the percentage of converging nodes. The first is that the percent of converging nodes will consider large positive orders of accuracy as asymptotic even though the value is far from the formal order of accuracy. The second reason is that the average distance from the formal order is more easily relatable to asymptotic behavior due to the reference to the order of accuracy.

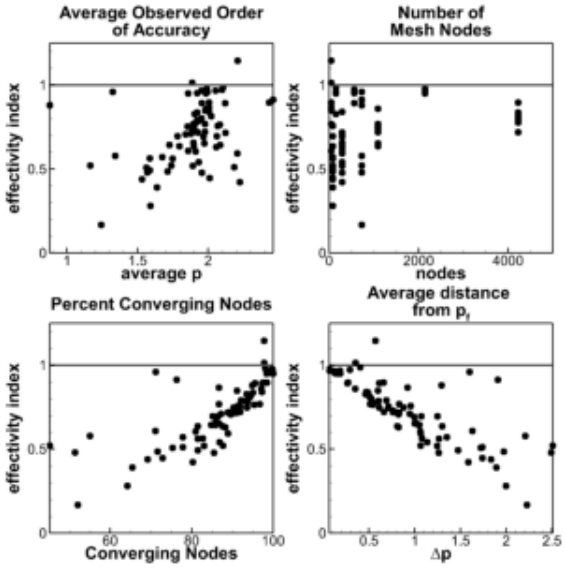


Figure 8.4. Various metrics considered to capture the asymptotic convergence of the data

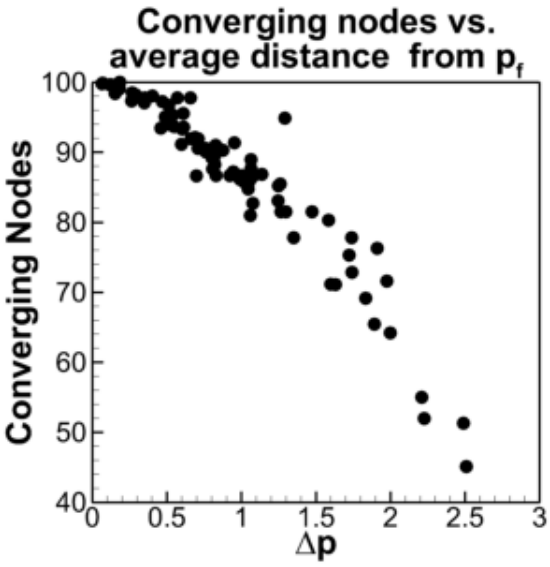


Figure 8.5. Comparison between the percent converging nodes versus the average distance from the formal order

8.4.1 Average distance from the formal order of accuracy

The average distance from the formal order of accuracy is calculated by

$$\Delta\overline{p} = \frac{1}{N}\sum_{i=1}^{N} min\left(\left|\hat{p}_i - p_f\right|, 2p_f\right). \tag{8.7}$$

where $\hat{p}$ is the observed order of accuracy calculated using Equation (3.2) or (3.3) and N is the total number of data points for the specific variable. The average distance from the formal order of accuracy is limited to twice the formal order (e.g. observed orders of accuracy of -4 and 8 would be limited to 4 for a formal order of accuracy of two. Observed orders of accuracy that are oscillatory are assigned a value of the maximum distance from the formal order, $2p_f$. The choice of twice the formal order as the maximum distance was empirically-based and tended to representation the asymptotic behavior of the data best.

8.4.2 Metric Curve-fitting

There are several data points evaluated for several different estimators which results in significant clutter when compared on the same figure. It is desired to further reduce this data for easy comparison between the various estimators which captures the asymptotic behavior of each respective estimator and allows for comparison between the distributions of the data. To capture the asymptotic behavior, a least squares curve-fit is calculated for each metric of each estimator, and to capture the scatter of data the 95 percent confidence bounds on the curve-fit is calculated. MATLAB's curve fitting toolbox [45] is used to calculate the curve-fit and the confidence bounds. A second-order polynomial is the basis-function for all fits and there were two different types of fits depending on the data being fit. The standard least squares fit is used to fit the conservativeness versus the average distance from the formal order where the confidence bounds contain 95 percent of the data. The effectivity index for error and uncertainty were fit using a "robust" least squares fit. A robust fit is an iterative fitting method where each data point is weighted depending on the error in the curve-fit for that specific data point. For a standard least squares fit, data points with a large error will affect the fit more and may tend to not capture the overall trend of the data. For a robust fit, the data points with a large error has a lower weighting than data points with a small error and will tend to capture the trend in the data. The reduced weighting also reduces the magnitude of the confidence bounds resulting in smaller bounds that may not be a 95 percent confidence bound on the data. The robust fit captures the asymptotic behavior of the data and represents the distribution more accurately than the standard least squares fit. An example is shown in Figure 8.6 comparing the effectivity index for two different uncertainty estimators illustrating the clear representation of the asymptotic behavior and the distribution in the data by the curve-fit bounds where the figure on the right has significantly less distribution in the data resulting in a more predictable estimate compared with the figure on the left.

The MATLAB code used to calculate both the standard and the robust fit is shown in Figure A3.6. The specific options for the different fits are listed in Table A3.4

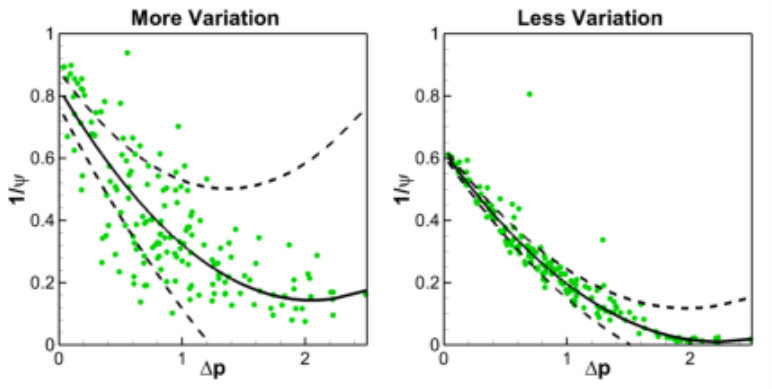


Figure 8.6. Fit example for two different uncertainty estimators showing more variation and less variation in the data represented by the curve-fit bounds

9 Results

The effectivity index for error and uncertainty, the conservativeness, and the average distance from the formal order of accuracy are calculated using Equations (8.1)-(8.4) for each primitive variable solution. Results for solution functionals discussed in the previous section are also included. A quadratic regression fit of the results versus the average distance from the formal order of accuracy is calculated along with the 95 percent confidence bounds similar to the example shown in Figure 8.6. For both the error and uncertainty effectivity index, the confidence bounds are shown to compare the asymptotic behavior of the estimators and the scatter of the data. For the conservativeness, only the lower confidence bound is shown. The results are shown in Figure 9.1. The error and uncertainty effectivity index are split into two figures for clarity where the lower figures show results for the LSQ methods with a refinement factor of two and 4/3. The LSQ methods show significantly different behavior between the choice of refinement factor due to the fourth grid level required for the estimate. The refinement factor between the finest and coarsest grids for an initial refinement factor of two is eight and for a refinement factor of 4/3 is only 2.6. This means that for a refinement factor of two, the coarsest mesh is significantly less asymptotic than the coarsest mesh used for a refinement factor of 4/3 and results in a completely different trend. For the other error and uncertainty estimators, there was no difference in trend between refinement factors of two and 4/3.

The inverse error effectivity index for the CF and FS methods are identical and are very similar to the GCI-OR method. The asymptotic behavior of these error estimators is similar to the GCI-glb method except the global average method has less scatter in the data. The GCI-2g error estimator is Richardson extrapolation using the formal order of accuracy and results in a large portion of the data with an inverse

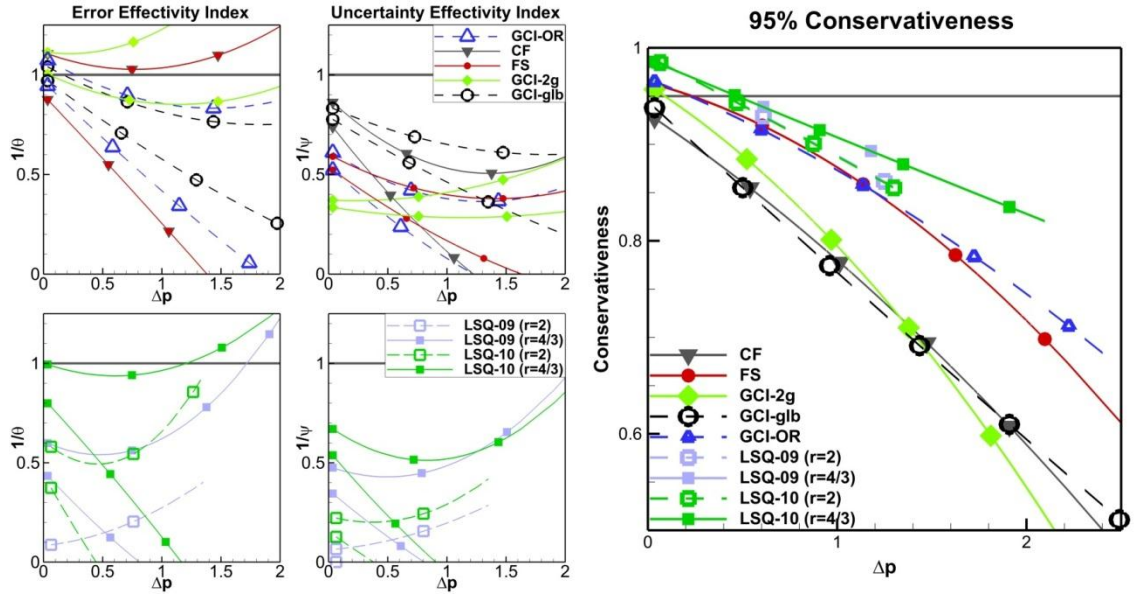


Figure 9.1. Error and uncertainty estimator results showing the 95 percent confidence bounds of the curve-fit for the effectivity index and the lower 95 percent confidence bound for the conservativeness

effectivity index greater than one meaning that the error estimate underestimates the true error. The LSQ methods show significantly different trends depending on the chosen refinement factor. The LSQ-09 error estimator with a refinement factor of two greatly overestimates the error which is improved by using a refinement factor of 4/3. The LSQ-10 error estimator shows a significant improvement for both refinement factors over the LSQ-09 error estimator, and specifically, the use of a refinement factor of 4/3 results in comparable error estimation compared to the other results.

For the uncertainty effectivity index, all but the GCI-2g uncertainty estimators have a similar trend which is larger scatter in the data with a lower inverse effectivity index away from the asymptotic range. As the solutions approach the asymptotic range, the scatter in the data decreases and the inverse effectivity index increases approaching the inverse of the factor of safety. The curve-fit bounds were representative of the data for all uncertainty estimators except GCI-OR. The data and the curve-fit bounds for this uncertainty estimator are shown in Figure 9.2. The trend in the GCI-OR changes at $\Delta\overline{p} = 0.2$ due to the discontinuous change in factor of safety when the observed order of accuracy is ten percent away from the formal order of accuracy. This change in trend could not be captured by a quadratic trend line. The behavior for the GCI-2g uncertainty estimator differs from the other uncertainty estimators because the inverse effectivity index is generally constant for all values of observed order of accuracy and only the scatter in the data decreases. The behavior of the inverse uncertainty effectivity index for the LSQ methods is similar to the corresponding inverse error effectivity index where the use of a refinement factor of 4/3 results in a more accurate estimate. The LSQ-10 uncertainty estimator outperforms the LSQ-09 estimator similar to the error estimator.

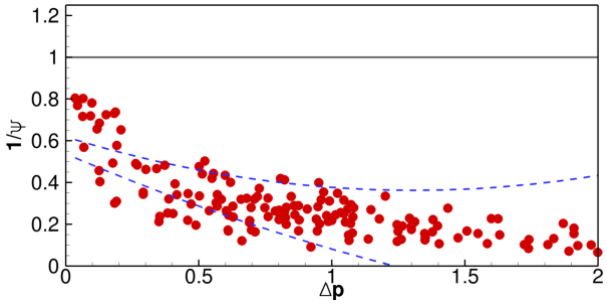


Figure 9.2. Uncertainty effectivity index fit bounds and data for GCI-OR

When comparing all of the inverse uncertainty effectivity index results, the GCI-glb estimator has the most accurate uncertainty estimates and the smallest scatter in the data. The CF method has similar asymptotic behavior, but the scatter in the data is much larger. Taking in to consideration the poor representation of the curve-fit bounds for the GCI-OR estimator, the asymptotic behavior is similar to that of the GCI-glb and CF estimators.

When comparing the conservativeness of all of the estimators, only the lower bound is shown. The target conservativeness for an uncertainty estimate is 95% which is shown in light grey as a reference. The most conservative uncertainty estimators are the LSQ methods for a refinement factor of 4/3. A refinement factor of two results in a slightly lower conservativeness for both LSQ methods. All other uncertainty estimators

fall short of reaching the 95 percent conservative goal for even moderate values of $\Delta\overline{p}$ with the GCI-glb, GCI-2g, and the CF estimators performing the worst. At the cost of a higher computational expense, the LSQ-10 uncertainty estimator outperforms all of the other estimators in terms of conservativeness when a refinement factor of 4/3 is used. If a refinement factor of two is used, the results are slightly less conservative but still outperform the other methods.

The effectivity index is a single global metric calculated from local data with no information provided about the distribution of the local data. PDFs are created for each of the error and uncertainty estimators to show the distribution of the inverse local effectivity index. Select PDFs for the supersonic manufactured solution are shown in Figure 9.3 and correspond to a single data point used to calculate the regression fit bounds shown in Figure 9.1. The first row is a set of PDFs for Richardson extrapolation calculated using the formal order of accuracy. The PDFs progress from a 65x65 mesh to the 513x513 mesh with a slight asymmetric distribution centered at one. The second row shows the inverse local uncertainty effectivity index for the GCI-OR, CF, FS, and LSQ-10 estimators. These 4 distributions provide some insight into the behavior of each estimator where the concentration of data for all except the LSQ-10 correspond to the respective factors of safety (e.g. the GCI-OR has peaks near 0.33 and 0.8 corresponding to the factors of safety of 3 and 1.25, respectively). The sign is taken into consideration for the uncertainty PDFs where a negative means that the corresponding error estimate was in the opposite direction of the true error. Most of the data falls between zero and one meaning that the uncertainty estimates are conservative and the corresponding error estimate has the correct sign.

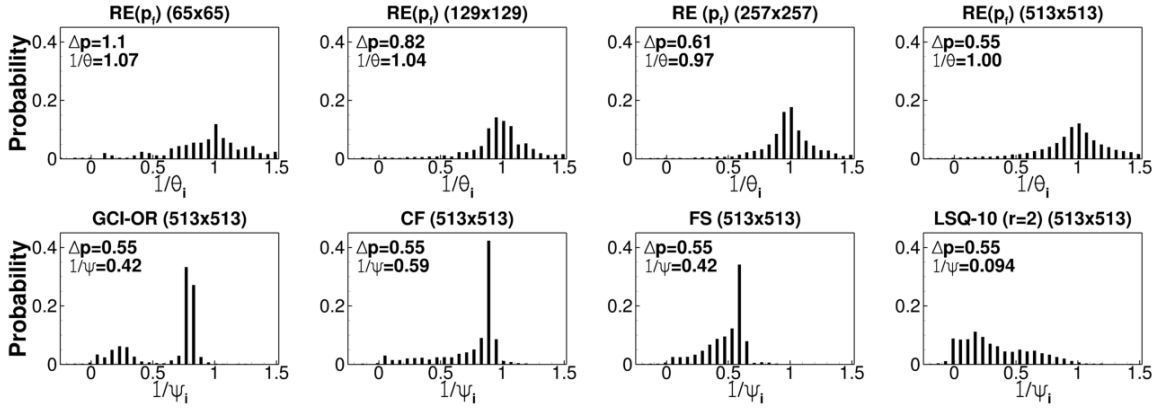


Figure 9.3. The local distribution of effectivity index for the supersonic manufactured solution density variable

The asymptotic convergence of local quantities is not independent of the rest of the solution. Error is transported through the solution domain making the convergence of local quantities noisier and the observed order of accuracy less representative of the asymptotic nature of the solution. Taking this into consideration, an important observation with respect to the use of observed order of accuracy can be made. The use of the global observed order of accuracy results in less scatter in the data and a more conservative uncertainty estimate. The GCI-OR uncertainty estimator is compared using different observed orders of accuracy in Figure 9.4. The orders of accuracy used are the

local observed order of accuracy (Section 3.2.2), the global average observed order of accuracy (Section 3.2.3), and the order of accuracy calculated from the average Δp

$$p = p_f - \Delta\overline{p}. \tag{9.1}$$

There is less scatter in the effectivity index when the globally averaged order of accuracy is used and there is an increase in conservativeness for solutions outside of the asymptotic range. When the average Δp is used, the scatter in the effectivity index decreases significantly and all solutions with a $\Delta\overline{p}$ less than 1.5 are at least 95 percent conservative (right side of Figure 9.4). This trend is observed for all of the GCI-based uncertainty estimators. The trade-off for less scatter in the inverse effectivity index and more conservativeness is a slightly lower inverse effectivity index. Taking this into consideration, local observed orders of accuracy should not be used to estimate discretization error and uncertainty.

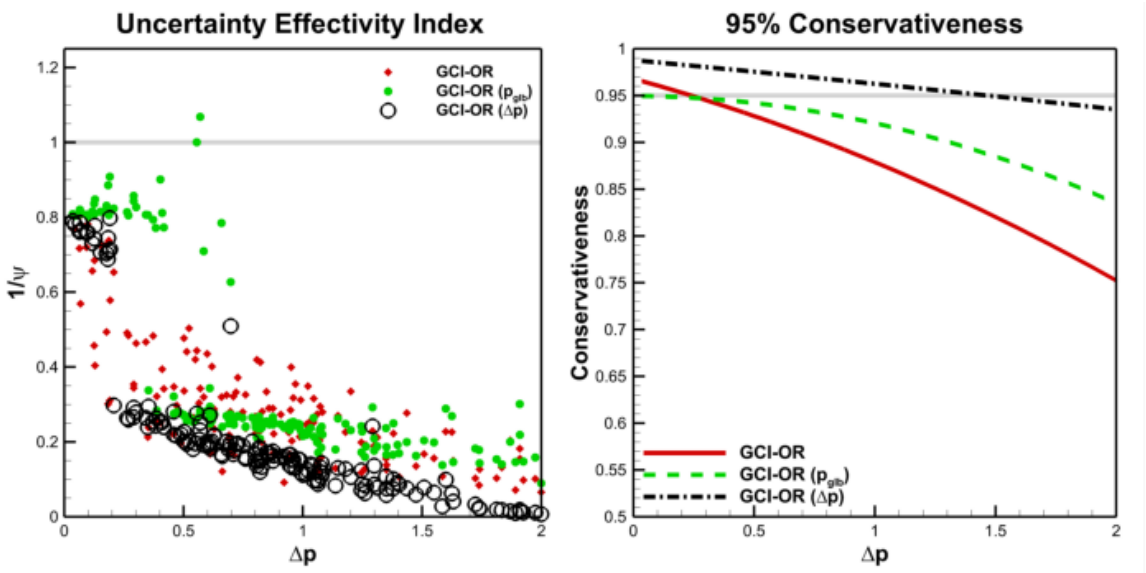


Figure 9.4. The GCI-OR uncertainty estimator with different observed orders of accuracy

9.1 Proposed Uncertainty Estimator

The use of $\Delta\overline{p}$ results in a more conservative estimate because observed orders of accuracy greater than the formal order are treated similarly to observed orders of accuracy less than the formal order. Based on these results, a discretization uncertainty estimator is proposed. The method used to calculate the average $\Delta\overline{p}$ for the results presented thus far specified a maximum $\Delta\overline{p}$ of 2 and set oscillating data points to this maximum value. Numerical experiments showed that results are highly sensitive to the choice of this parameter that would require either an arbitrary specification or calibration. Observed order of accuracy cannot be calculated for oscillating data points to investigate the corresponding behavior of error estimation. Instead, a term related to the numerator of the observed order of accuracy equation is used to quantify solution convergence referred to as the convergence ratio: $(f_2 - f_1)/(f_3 - f_2)$. A convergence ratio between 0 and 1 corresponds to a positive observed order of accuracy. For a formal order of accuracy of two, the convergence ratio is 0.25. A convergence ratio greater than one corresponds to a

negative observed order of accuracy, and a negative convergence ratio corresponds to oscillatory data (i.e. undefined observed order of accuracy). For all local error estimates with a refinement factor of 2, the inverse local effectivity index for Richardson extrapolation using the formal order of accuracy is plotted versus the convergence ratio in Figure 9.5. The observed order of accuracy calculated using Equation (3.2)

$$\hat{p} = \frac{ln\left(\frac{f_3 - f_2}{f_2 - f_1}\right)}{ln(r)} \tag{3.2}$$

versus convergence ratio is shown as a solid line and approaches infinity as the difference between the fine grid solution and the medium grid solution approach zero. The dashed line is a reference line with an effectivity index of one. All data points should approach a convergence ratio of 0.25 and an effectivity index of one with mesh refinement for a formal order of accuracy of two. The distribution of the data is symmetric about the vertical axis at a convergence ratio of zero which suggests that there is equivalent uncertainty for positive and negative convergence ratios. Even though the data with a negative convergence ratio is oscillating and the observed order of accuracy cannot be calculated, the majority of the discretization error estimates have a positive convergence ratio meaning that the sign of the error is accurate and is near one meaning that the magnitude of the estimate is accurate. A more quantitative look at the distributions of the inverse effectivity index is made by creating probability density functions for three different convergence ratios to compare the distribution of (inverse) effectivity index at positive and negative convergence ratios. Figure 9.6 shows PDFs for convergence ratios of ± 0.1, ± 0.25, and ± 1.0. The PDFs compare favorably for the first two convergence ratios. The latter convergence ratio has misaligned peaks with values at 0.3 and 0.1 for the positive and negative convergence ratio, respectively. The accuracy of the error estimates at the positive convergence ratio is not significantly more accurate and the two PDFs can be considered comparable. As the convergence ratio approaches zero, the

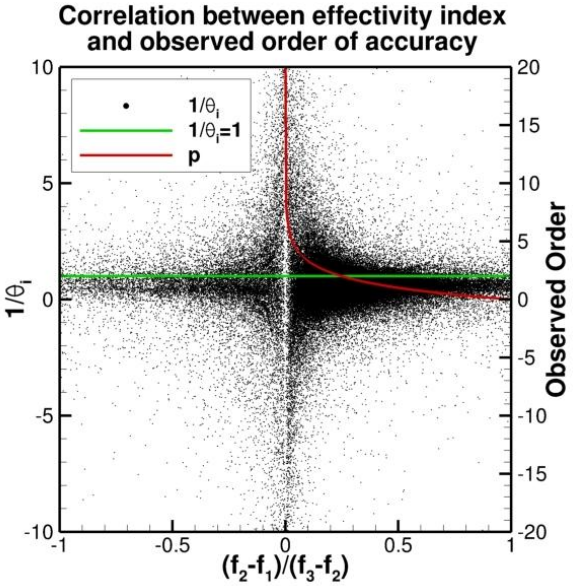


Figure 9.5. Inverse local effectivity index versus solution convergence

inverse effectivity index approaches positive and negative infinity meaning that the error estimate is grossly underestimated for large observed orders of accuracy. Figure 9.5 implies that the largest uncertainty associated with Richardson extrapolation error estimators is large observed orders of accuracy and that oscillating data has a similar distribution of accuracy as non-oscillating data.

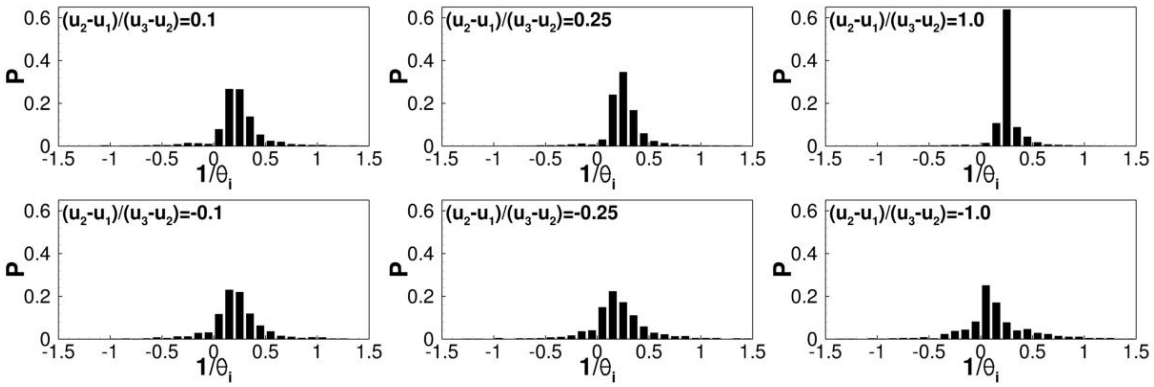


Figure 9.6. Probability distribution functions of inverse local effectivity index taken at positive and negative convergence ratios

Based on these conclusions, the observed order of accuracy for the proposed uncertainty estimator is modified from Equation (3.2) using an absolute value

$$\hat{p} = \frac{ln\left(\left|\frac{u_{r^2h} - u_{rh}}{u_{rh} - u_h}\right|\right)}{ln(r)}. \tag{9.2}$$

In Equation (8.7), the observed order of accuracy was limited to $2p_f$; however, this limiting term is removed from the proposed uncertainty estimator. Instead the average Δp is calculated using

$$\Delta\overline{p} = \frac{1}{N}\sum_{i=1}^{N}\left|\hat{p}_i - p_f\right|. \tag{9.3}$$

and the observed order of accuracy is

$$p = p_f - min\left(\Delta\overline{p}, 0.95p_f\right). \tag{9.4}$$

The minimum observed order of accuracy for use in an uncertainty estimate is 0.1 for a formal order of accuracy of two. This parameter is subject to calibration and was chosen in favor of a more conservative error estimate because an average Δp greater than 1.9 means that the solutions are very far from the asymptotic range. The factor of safety is a function of the order of accuracy and is specified by the equation

$$FS(p) = \left[F_0 - (F_0 - F_1)\left(\frac{p}{p_f}\right)^8\right] \tag{9.5}$$

where $F_0 = 3.0$ and $F_1 = 1.1$ prescribe the behavior of the factor of safety as the order of accuracy varies. The factor of safety is 3 far from the formal order of accuracy and approaches 1.1 when the order of accuracy approaches the formal order of accuracy. The

order of accuracy of 1.5 is about the value where the data begins to show asymptotic behavior with more reliable error estimates. The exponent 8 was chosen so that the factor of safety remains 3 for orders of accuracy ranging from 0 to about 1.5 and begins to decrease to 1.1 for orders of accuracy greater than 1.5. This factor of safety versus order of accuracy as computed from Equation (9.5) is shown in Figure 9.7. The corresponding error estimator is Richardson extrapolation using the formal order of accuracy, Equation (3.2), and is chosen because the effectivity index is near one for the widest range of average Δp. The final uncertainty estimator is

$$U(p) = FS(p)\left(\frac{r^{p_f} - 1}{r^p - 1}\right)|\bar{\varepsilon}_h|. \tag{9.6}$$

The middle term in Equation (9.6) is to compensate for not using the observed order of accuracy in the error estimator. The method used to evaluate the uncertainty estimators discussed in this paper is applied to the proposed estimator. The resulting 95 percent confidence bounds for the inverse error effectivity index, inverse uncertainty effectivity index, and conservativeness are shown in Figure 9.7 for all cases examined herein. All regression fits are calculated in the same manner as those shown in Figure 8.6. The 95 percent confidence bounds for the Factor of Safety method are also shown for reference. The lower bound for conservativeness is 95 percent conservative through the entire range of order of accuracy (including some very coarse grids that are well outside the asymptotic range). The resulting uncertainty effectivity index is slightly lower than the factor of safety method but the scatter of the data is significantly lower.

The proposed estimator is further evaluated using numerical solutions which were not considered during development. Additional numerical solutions were computed using Loci-Chem [46], a 3D, unstructured flow solver developed at Mississippi State University for chemically reacting flows. Loci-Chem has undergone significant code verification using the Method of Manufactured Solutions [47]. The proposed estimator is applied to several of these manufactured solutions which include the Euler and Navier-Stokes equations. The manufactured solutions for the Navier-Stokes equations also include the baseline-kε and baseline-kω turbulence models, and extrapolation, farfield, and inflow boundary conditions. All manufactured solutions are computed on hexahedral, triangular prismatic, and tetrahedral grid topologies with grid quality varying from a cube with uniform spacing to a highly skewed curvilinear grid for a total of about 100 additional error and uncertainty estimates. The inverse error effectivity index, inverse uncertainty effectivity index, and conservativeness are shown in Figure 9.7. All results for the Loci-Chem solutions match the trends for the 95 percent confidence bounds well. There is slightly more scatter in both of the effectivity indexes; however, all but one estimate is at least 95 percent conservative.

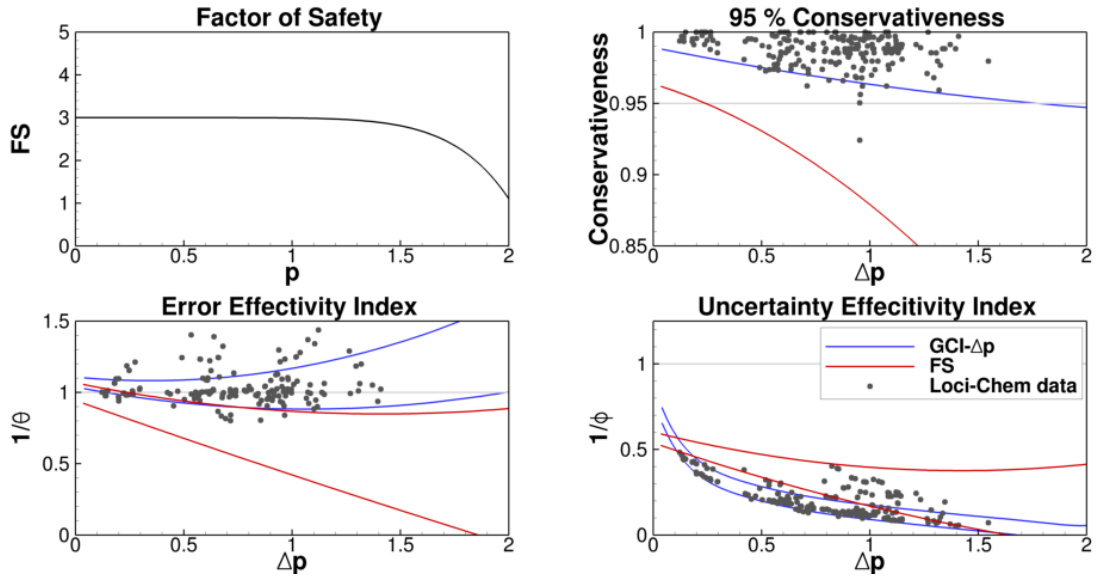


Figure 9.7. Error and uncertainty metrics for the proposed estimator compared to the Factor of Safety Method

10 Conclusions

Richardson extrapolation-based discretization error and uncertainty estimators were applied to six different cases, four different dependent variables, and global quantities from multiple grid levels. The reliability of the error and uncertainty estimates was quantified using the effectivity index and the overall conservativeness of the estimate. The resulting metrics were combined using the average distance from the formal order of accuracy for each data point which was able to accurately capture the asymptotic behavior of each estimator. The resulting data was fitted with a least squares quadratic polynomial with 95 percent confidence bounds to quantify the scatter in the data. Overall there was a general trade-off between the accuracy of the error and uncertainty estimates and the conservativeness. The most conservative estimators, the LSQ methods, greatly overestimated the error, and the most accurate estimators, the global averaging method and correction factor method, were the least conservative. The Factor of Safety method and the Oberkampf and Roy GCI method offered trade-offs between conservativeness and accuracy.

Two different LSQ methods were used with two different refinement factors. The effectivity index and the conservativeness were highly sensitive to the choice of refinement factor where a refinement factor of 4/3 showed significant improvements over a refinement factor of 2. The LSQ-10 method outperformed the LSQ-09 method in every metric except conservativeness which was nearly indistinguishable. The LSQ method has the disadvantage of requiring four solutions instead of three which can be considerably more expensive depending on the size of the problem.

There were also significant differences when comparing the use of local and global orders of accuracy for the same estimator. The use of a global order of accuracy significantly reduces the scatter in effectivity index while also greatly improving the conservativeness of the estimate as shown in Figure 9.4. Based on these results, it is recommended that local observed orders of accuracy not be used to estimate error or uncertainty.

The use of the average distance from the formal order of accuracy represented the asymptotic convergence of the estimators very well, so an uncertainty estimator was proposed which calculates the order of accuracy of the solution using this metric. For the cases examined, the majority of uncertainty in an error estimate comes from vanishingly small differences between the fine and medium solutions relative to the difference between the medium and coarse solutions as opposed to oscillatory convergence. The observed order of accuracy was thus modified to include an absolute value instead of prescribing arbitrary treatment of oscillating data points. The proposed estimator was developed from solutions consisting of primarily the Euler equations. Additional solutions computed using a 3D, unstructured, finite-volume flow solver were used to test the proposed estimator where more than 99 percent of the Loci-Chem uncertainty estimates were at least 95 percent conservative.

11 References

1. Schetz, J., *Boundary Layer Analysis*, Pretence-Hall, Upper Saddle River, NJ, 1993.
2. Priebe, S., "Martin, M. P., "Direct Numerical Simulation of Shockwave and Turbulent boundary Layer Interactions," Presented at the 47th AIAA Aerospace Sciences Meeting Including The New Horizons Forum and Aerospace Exposition, 5-8 January 2009, Orlando, Florida.
3. Oberkampf, W. L., Roy, C. J., *Verification and Validation in Scientific Computing*, Cambridge University Press, New York City, NY.
4. Roy, C. J., "Review of Code and Solution Verification Procedures for Computational Simulation," *Journal of Computational Physics*, Vol. 206, Issue 1, pp. 131-156.
5. *IEEE Standard for Floating-Point Arithmetic*, New York, Microprocessor Standards Committee, Institute of Electrical and Electronics Engineers Computer Society, 2008.
6. Pereyra, V. "On Improving an Approximate Solution of a Functional Equation by Deferred Corrections," *Numerische Mathematik* , Vol. 8, 1965, pp. 376-391.
7. Stetter, H. J. "The Defect Correction Principle and Discretization Methods," *Numerische Mathematik*, Vol. 29, 1978, pp. 425-443.
8. Babushka, I. and Rheinboldt, W. C., "A Posteriori Error Estimates for the Finite Element Method," *International Journal for Numerical Methods in Engineering*, Vol 12, pp. 1597-1615.
9. Jameson, A., "Aerodynamic Design via Control Theory," *Journal of Scientific Computing*, Vol. 3, No. 3, pp.233-260.
10. Fehlberg, E., "Low-Order Classical Runge-Kutta Formulas with Step Size Control and their Applications to some Heat Transfer Problems," NASA Technical Report 315, National Aeronautics and Space Administration, July 1969.
11. Richardson, L. F., "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*, Series A, Vol. 210, 307-357, 1910.
12. Richardson, L. F., "The Deferred Approach to the Limit," *Transactions of the Royal Society of London, Series A*, Vol. 226, pp. 299-361, 1927.
13. Roache, P. J., "*Computational Fluid Dynamics*, First Edition, Hermosa Publishers, Albuquerque, New Mexico, 1972.
14. Roache, P. J., "Perspective: A Method for Uniform Reporting of Grid Refinement Studies," *ASME Journal of Fluids Engineering*, Vol 116, pp. 405-413.
15. Roache, P. J., "Verification of Codes and Calculations," AIAA Paper 95-2224, 26th AIAA Fluid Dynamics Conference, San Diego, California, June 19-22, 1995.
16. Roache, P. J., "Verification of Codes and Calculations," *AIAA Journal*, Vol. 36, No. 5, May 1998, pp. 696-702.
17. Roache, P. J. *Verification and Validation in Computational Science and Engineering*, Hermosa Publishers, Albuquerque, NM, 1998.
18. Roache, P. J., "Error Bars for CFD," Presented at the 41st Aerospace Sciences Meeting and Exhibit, 6-9 January 2003, Reno, Nevada.

19. Cadafalch, J., Perez-Segarra, C. D., Consul, R., Oliva, A., "Verification of Finite Volume Computations on Steady-State Fluid Flow and Heat Transfer," *Journal of Fluids Engineering*, Vol. 124, pp. 11-21.
20. Stern, F., Wilson, R., Coleman, H., Paterson, E., "Comprehensive Approach to Verification and Validation of CFD Simulations -- Part 1: Methodology and Procedures," *Journal of Fluids Engineering*, Vol. 123, No. 4, pp. 793-802.
21. Wilson, R., Stern, F., Coleman, H., Paterson, E., "Comprehensive Approach to Verification and Validation of CFD Simulations -- Part 2: Application for RANS Simulation of a Cargo/Container Ship," *Journal of Fluids Engineering*, Vol. 123, No. 4, pp. 803-810.
22. Wilson, R., Shao, J., Stern, F., "Discussion: Criticisms of the "Correction Factor" Verification Method," *Journal of Fluids Engineering*, Vol. 126, No. 4, pp.704-706.
23. Roache, P., " Criticisms of the "Correction Factor" Verification Method," *Journal of Fluids Engineering*, Vol. 125, No. 4, pp. 732-733.
24. Xing, T., Stern, F., "Factors of Safety for Richardson Extrapolation," IIHR Technical Report No. 469, March 2009.
25. Xing, T., Stern F., "Factors of Safety for Richardson Extrapolation," *Journal of Fluids Engineering*, Vol. 132, pp.
26. Eca, L., Hoekstra, M., "An Evaluation of Verification Procedures for CFD Applications," Presented at the 24th Symposium on Naval Hydrodynamics, 8-13 July, 2002.
27. Eca, L., Hoekstra, M., "Evaluation of Numerical Error Estimation Based on Grid Refinement Studies with the Method of Manufactured Solutions," *Computer and Fluids*, Vol. 38, No. 8, pp. 1580-1591.
28. Eca, L, "Uncertainty Quantification for CFD," Personal communication, April 7th, 2010, pp. 105-132.
29. Satav, V., Hixon, R., Nallasamy, M., Sawyer, S., "Validation of a Computational Aeroacoustics Code for Nonlinear Flow about Complex Geometries Using Ringleb's Flow," Presented at the 11th AIAA/CEAS Aeroacoustics Conference, 23-25 May 2005, Monterey, California
30. Ollivier-Gooch, C., Nejat, A., Michalak K., "Obtaining and Verifying High-Order Unstructured Finite Volume Solutions to the Euler Equations," *AIAA Journal*, Vol. 47, No. 9, 2009, pp. 2105-2120.
31. Wilcox, D. C., *Turbulence Modeling for CFD*, 2nd ed., DCW Industries, La Canada, CA, 1998.
32. Menter, F. R., "Improved Two-Equation k-omega Turbulence Models for Aerodynamic Flows," NASA TM 103975, October 1992.
33. ANSYS FLUENT 12.0, Theory Guide, 2009.
34. Roy, C. J., Nelson, C. C., Smith, T. M., Ober, C. C., "Verification of Euler/Navier-Stokes Codes using the Method of Manufactured Solutions," *International Journal of Numerical Methods in Fluids*, Vol. 44, No. 6, 2004, pp. 599-620.
35. Roe, P. L. "Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes," *Journal of Computational Physics*, Vol. 135, 1997, pp. 250-258.
36. Van Leer, B. "Towards the Ultimate Conservative Difference Scheme. V. A Second order Sequel to Godunov's Method." *Journal of Computational Physics,* Vol. 54, pp. 174-201.

37. Leonard, B. P. “A Stable and Accurate Convective Modeling Procedure Based on Quadratic Upstream Interpolation,” *Computer Methods of Applied Mechanics and Engineering*, Vol. 19, pp. 59-98.
38. Tecplot 360 version 13.1.1.16314, Bellevue, Washington: Tecplot Inc., 2011.
39. Rumsey, C. L., Smith, B. R., Huang, G. P., “Description of a Website Resource for Turbulence Modeling Verification and Validation,” Presented at the 40th AIAA Fluid Dynamics Conference and Exhibit, June 28-July 1, 2010, Chicago, IL.
40. Langley Research Center, "FUN3D-analysis and design," http://func3d.larc.nasa.gov. November 27th, 2011.
41. Langley Research Center, "CFL3D Version 6.6," http://cfl3d.larc.nasa.gov. November 27th, 2011.
42. Rumsey, C. L., Thomas, J. L., “Application of FUN3D and CFL3D to the Third Workshop on CFD Uncertainty Analysis,” NASA/TM-2008-215537
43. Banks, J. W., Aslam, T., Rider, W., J., “On Sub-linear Convergence for Linearly Degenerate Waves in Capturing Schemes,” *Journal of Computational Physics*, Vol. 227, Issue 14, pp.6985-7002.
44. Ainsworth M., Oden, J. T., A Posteriori Error Estimation in Finite Element Analysis, Wiley-interscience. 2000.
45. MATLAB version 7.12.0.635, Natick, Massachusetts: The MathWorks Inc., 2011.
46. Luke, E. A., Tong, X., Wu, J., Cinnella, P., "Chem 3.2: A Finite-Rate Viscous Chemistry Solver -- The User Guide," Tetra Research Corporation, 2010.
47. Veluri, S. P., Code verification and numerical accuracy assessment for finite volume CFD codes, PhD dissertation, Virginia Tech, Blacksburg, Virginia, 2010.

Appendix 1. Observed Order of Accuracy Lower Limit

Richardson extrapolation is accurate only when all three solutions used to estimate discretization error are asymptotic. However, in real applications asymptotic solutions can be difficult to obtain. Error estimates are still desired for non-asymptotic solutions which result in oscillatory or diverging convergence or very slow convergence with observed orders of accuracy near zero. Several error estimators use the observed order of accuracy. As the observed order of accuracy approaches zero the normalized Richardson extrapolation approaches infinity as shown in Figure A1.1. Normalized Richardson extrapolation is the Richardson extrapolation error estimate using the observed order of accuracy divided by the Richardson extrapolation error estimate using the formal order of accuracy

$$\bar{\varepsilon}_n = \frac{\frac{f_2 - f_1}{r^{\hat{p}} - 1}}{\frac{f_2 - f_1}{r^{p_f} - 1}} = \frac{r^{\hat{p}} - 1}{r^{p_f} - 1}. \qquad \text{(A1.1)}$$

To provide a reasonable error estimate a lower limit should be placed on the observed order of accuracy. In addition, the lower limit should also provide a reasonable error estimate for oscillatory or diverging convergence.

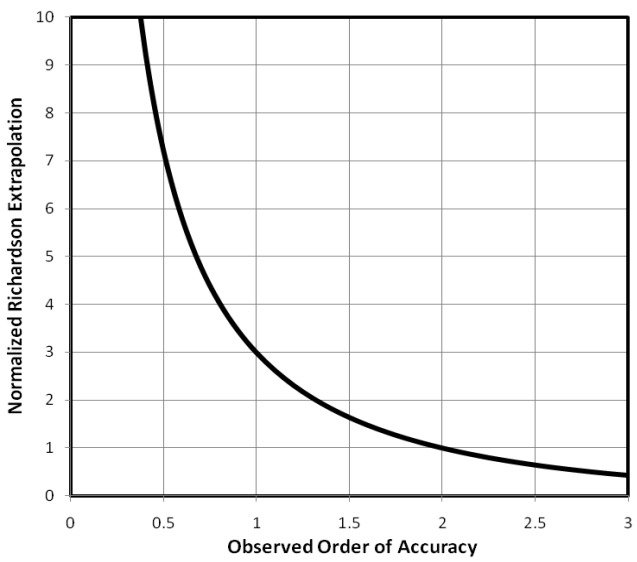


Figure A1.1. Normalized Richardson extrapolation versus observed order of accuracy

To determine an appropriate lower limit, five different bins are created to classify the type of convergence, see Table A1.1. Five different values for the lower limit p_l are investigated ranging from 0.1 to 0.5. To determine the most appropriate lower limit, discretization error and uncertainty is estimated using four different solutions and evaluated using conservativeness and the error effectivity index metrics. Specifically, the estimator used is the GCI-2g method.

Table A1.1. Solution convergence bins

Convergence	Order
Oscillatory	$\hat{p} = undefined$
Diverging	$\hat{p} < 0$
Converging	$0 < \hat{p} < p_l$
Converging	$p_l < \hat{p} < p_f$
Converging	$p_f < \hat{p}$

A1.1. Solutions

Two of the four solutions used to determine a lower limit or discussed in Chapter 5. These solutions are the supersonic manufactured solution and Ringleb's flow. The other two solutions consist of a more complicated manufactured solution and a case with an oblique shock. The latter two cases are briefly described in the following two sections.

A1.1.1. Complex Manufactured Solution

The more complex manufactured solution is identical to the manufactured solution cases discussed in Chapter 5 except the coefficients were changed to increase the curvature of the solution. The coefficients are shown in Table A1.2. The solutions are solved using the first mesh family with a refinement factor of two. A sample solution is shown in Figure A1.2.

Table A1.2 Summary of coefficients for the supersonic manufactured solution

	a_0	a_x	a_y	s_x	s_y
$\rho(x,y)$	1.0	0.15	-0.1	8.0	4.0
$u(x,y)$	800.0	50.0	-30.0	10.0	5.0
$v(x,y)$	800.0	-75.0	40.0	4.0	5.0
$p(x,y)$	1.0E5	2.0E4	5.0E4	8.0	4.0

A1.1.2. Oblique Shock

An oblique shock case is included as a challenge problem due to the strong singularity. The Mach 2.9 flow is impinging on a flat plate with a deflection angle of ten degrees. The static pressure and temperature are 10,000 Pa and 200.0 K, respectively. The finest mesh is 513x513 and is refined by a factor of two to create several mesh levels where the coarsest mesh is 17x17. A sample solution is shown in Figure A1.3.

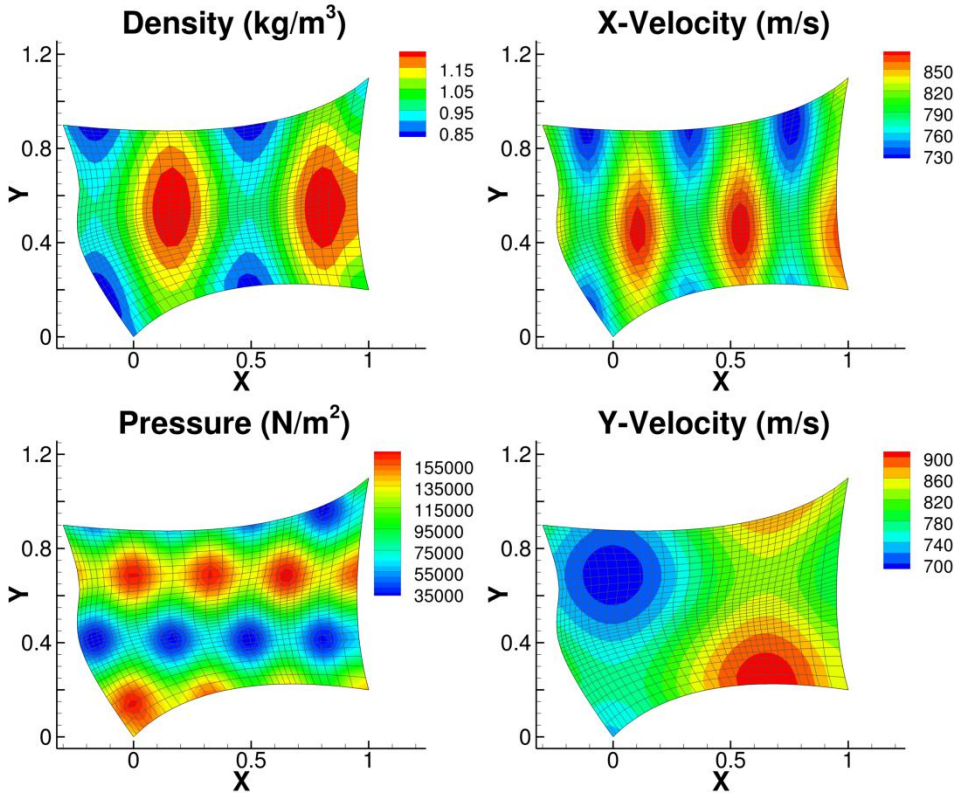


Figure A1.2. Complex supersonic manufactured solution primitive variables with 33x33 mesh

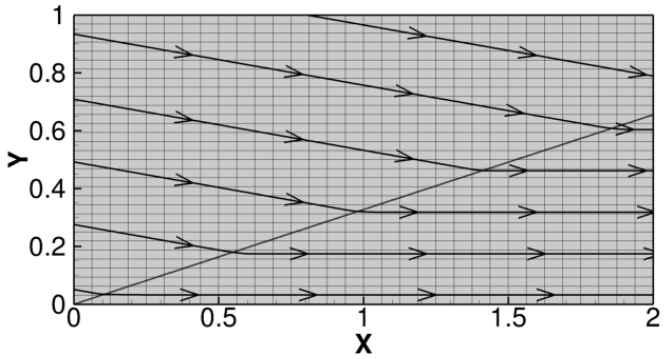


Figure A1.3. Oblique shock with 33x33 mesh and streamlines

A1.2. Lower Limit Results

The conservativeness and error effectivity index versus the chosen lower limit for each of the four cases discussed is shown in Figure A1.4. For all cases there is a very clear trend. The conservativeness of the estimator remains almost constant as the lower limit is varied from 0.1 to 0.5. Only for the oblique shock case does the conservativeness have a noticeable decline. For the effectivity index, all cases show a substantial decrease as the lower limit is increased to 0.5.

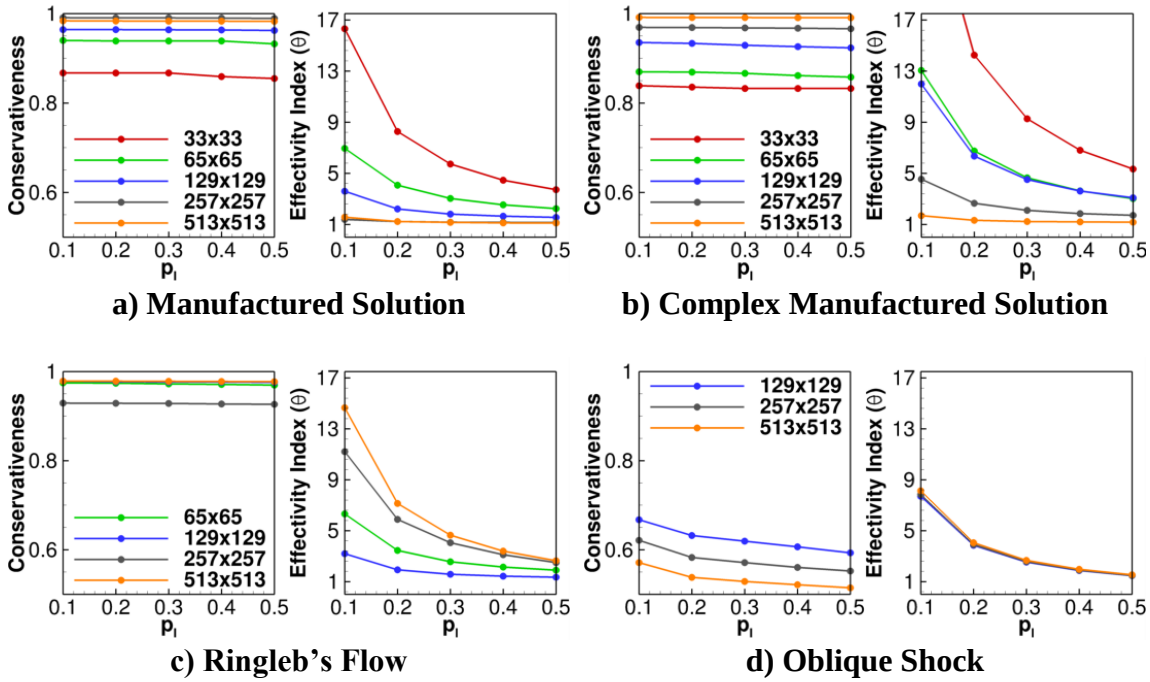


Figure A1.4. Conservativeness and error effectivity index for the manufactured solution, complex manufactured solution, Ringleb's flow, and an oblique shock

If the same metrics are considered for each bin shown in Table A1.1, the trends are the same except for one. The conservativeness is most affected when the observed order of accuracy is undefined (shown in Figure A1.5). There is a significant decrease in conservativeness as the lower limit is increased; however, undefined orders of accuracy make up a relatively small fraction of the total data and affect the total conservativeness very little. See Table A1.3 for the percentage of data in each bin.

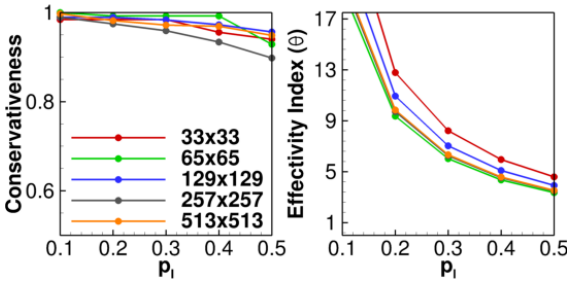


Figure A1.5. Conservativeness and error effectivity index for the oscillating nodes for the manufactured solution

Table A1.3. Percentage of data in each bin for the manufactured solution case

bin	33x33	65x65	129x129	257x257	513x513
$0 < \hat{p} < p_l$	1.23	0.41	0.26	0.24	0.07
$\hat{p} < 0$	7.07	4.93	1.78	1.47	0.95
$\hat{p} = und$	30.73	12.02	5.06	4.64	2.35
$p_l < \hat{p} < p_f$	36.53	55.34	64.00	75.16	55.85
$p_f < \hat{p}$	25.26	27.44	31.17	22.73	40.78

Based on the lack of sensitivity of conservativeness to the lower limit on observed order of accuracy and the significant decrease in error effectivity index for an increasing lower limit, the optimal lower limit is 0.5. The lower limit is used for diverging and oscillating data points in addition to limiting orders of accuracy below the specified lower limit.

Appendix 2. Supersonic Vortex Flow FLUENT Details

A2.1. FLUENT Setup

The FLUENT setup for the supersonic vortex flow verification case is shown below. Options that are not relevant to the solution are omitted for brevity.

FLUENT
Version: 2d, dp, dbns imp (2d, double precision, density-based implicit, inviscid)
Release: 13.0.0

Model

Model	Settings
Space	2D
Time	Steady
Viscous	Inviscid
Heat Transfer	Enabled

Material Properties

Material: air (fluid)

Property	Units	Method	Value(s)
Density	kg/m3	ideal-gas	#f
Cp (Specific Heat)	j/kg-k	constant	1006.43
Molecular Weight	kg/kgmol	constant	28.966
Thermal Expansion Coefficient	1/k	constant	0
Speed of Sound	m/s	none	#f

Boundary Conditions

Name	id	type
pressure-inlet-5	5	pressure-inlet
wall-4	4	wall
pressure-outlet-6	6	pressure-outlet

Setup Conditions

pressure-inlet-5

Condition	Value

```
------------------------------------------------------------
Gauge Total Pressure (pascal)        100000
Supersonic/Initial Gauge Pressure   (profile udf p_profile)
    Total Temperature               (profile udf T0_profile)
    Direction Specification Method       1

pressure-outlet-6
Condition                           Value
------------------------------------------------------------------------
Gauge Pressure                      (profile udf p_profile)
Backflow Total Temperature          (profile udf T0_profile)
```

Solver Settings

```
---------------
Variable   Scheme
-----------------------------
Flow       Second Order Upwind

Time Marching
Parameter        Value
-------------------------
Solver           Implicit
Courant Number   1000000
```

A2.2. User Defined Function

The UDF used to define the pressure profile, temperature profile, and initial conditions is shown below.

```
/************************************************************************
  UDF for initializing flow field variables
************************************************************************/

#include "udf.h"

DEFINE_PROFILE(p_profile,t,i)
{
 face_t f;
 real x[ND_ND];

 real mi,ri,ro,rhoi,gamma,u,v,uloc,vloc,r,rho,P,Ploc,U,Ui,Umag,
 R,rhat,mweight,a,P0,T0,T,M;

 gamma=1.4;
 mi = 2.0;
 ri = 2.0;
 ro = 3.0;
 rhoi = 1.0;
 mweight = 28.966;
 rhat = 8314.4621;
```

```
 P0=100000;
 R = rhat/mweight;

  /* loop over all faces  */
  begin_f_loop(f,t)
      {
       F_CENTROID(x,f,t);
       r = sqrt(pow(x[0],2)+pow(x[1],2));
       Ui = mi*pow(rhoi,(gamma-1)/2);
       U = Ui*ri/r;

       rho = rhoi*pow((1.0+(gamma-1.0)/2.0*pow(mi,2)*(1.0-pow(ri,2)/pow(r,2))),(1.0/(gamma-1)));
       u = x[1]*U/r;
       v = -x[0]*U/r;
       Ploc = pow(rho,gamma)/gamma;

      Umag = sqrt(pow(u,2)+pow(v,2));
      a = sqrt(gamma*Ploc/rho);
      M = Umag/a;

      P = pow( 1.0+(gamma-1)/2*pow(M,2) , -gamma/(gamma-1.0) )*P0;
      a = sqrt(gamma*P/rho);
      T = P/(R*rho);
      T0 = ( 1.0+(gamma-1)/2*pow(M,2))*T;

      uloc=M*a*u/Umag;
      vloc=M*a*v/Umag;

      F_PROFILE(f,t,i) = P;
   }
 end_f_loop(f,t)
}



DEFINE_PROFILE(T0_profile,t,i)
{
 face_t f;
 real x[ND_ND];

 real mi,ri,ro,rhoi, gamma, u,v,uloc,vloc,r,rho,P,Ploc,U,Ui,Umag,R,rhat,mweight,a,P0,T0,T,M;


 gamma=1.4;
 mi = 2.0;
 ri = 2.0;
 ro = 3.0;
 rhoi = 1.0;
 mweight = 28.966;
 rhat = 8314.4621;
 P0=100000.0;
 R = rhat/mweight;


  /* loop over all cells  */
  begin_f_loop(f,t)
```

```
      {
       F_CENTROID(x,f,t);
       r = sqrt(pow(x[0],2)+pow(x[1],2));
       Ui = mi*pow(rhoi,(gamma-1)/2);
       U = Ui*ri/r;

       rho = rhoi*pow((1.0+(gamma-1.0)/2.0*pow(mi,2)*(1.0-pow(ri,2)/pow(r,2))),(1.0/(gamma-1)));
       u = x[1]*U/r;
       v = -x[0]*U/r;
       Ploc = pow(rho,gamma)/gamma;

      Umag = sqrt(pow(u,2)+pow(v,2));
      a = sqrt(gamma*Ploc/rho);
      M = Umag/a;


      P = pow( 1.0+(gamma-1)/2*pow(M,2) , -gamma/(gamma-1.0) )*P0;
      a = sqrt(gamma*P/rho);
      T = P/(R*rho);
      T0 = ( 1.0+(gamma-1)/2*pow(M,2))*T;

      uloc=M*a*u/Umag;
      vloc=M*a*v/Umag;


      F_PROFILE(f,t,i) = T0;
   }
  end_f_loop(f,t)
}


DEFINE_INIT(initialize,d)
{
  cell_t c;
  Thread *t;
  real x[ND_ND];

  real mi,ri,ro,rhoi, gamma, u,v,uloc,vloc,r,rho,rho0,P,Ploc,U,Ui,Umag,R,rhat,mweight,a,P0,T0,T,M;


  gamma=1.4;
  mi = 2.0;
  ri = 2.0;
  ro = 3.0;
  rhoi = 1.0;
  mweight = 28.966;
  rhat = 8314.4621;
  P0=100000.0;
  R = rhat/mweight;

  thread_loop_c(t,d)
  {

    /* loop over all cells  */
   begin_c_loop_all(c,t)
       {
```

```
        C_CENTROID(x,c,t);
        r = sqrt(pow(x[0],2)+pow(x[1],2));
        Ui = mi*pow(rhoi,(gamma-1)/2);
        U = Ui*ri/r;

        rho = rhoi*pow((1.0+(gamma-1.0)/2.0*pow(mi,2)*(1.0-pow(ri,2)/pow(r,2))),(1.0/(gamma-1)));
        u = x[1]*U/r;
        v = -x[0]*U/r;
        Ploc = pow(rho,gamma)/gamma;

       Umag = sqrt(pow(u,2)+pow(v,2));
       a = sqrt(gamma*Ploc/rho);
       M = Umag/a;


       P = pow( 1.0+(gamma-1)/2*pow(M,2) , -gamma/(gamma-1.0) )*P0;
       rho0 = rhoi*pow(1.+(gamma-1.)/2.*pow(mi,2),1./(gamma-1.) );

       a = sqrt(gamma*P/rho);
       T = P/(R*rho);
       T0 = ( 1.0+(gamma-1)/2*pow(M,2))*T;

       uloc=M*a*u/Umag;
       vloc=M*a*v/Umag;

       C_R(c,t) = rho;
       C_P(c,t) = P;
       C_U(c,t) = uloc;
       C_V(c,t) = vloc;
       C_T(c,t) = T;
    }
  end_c_loop_all(c,t)

  }
}
```

Appendix 3. Curve-fitting Details for Data Visualization

A3.1. MATLAB Curve-fitting

```
45      %% fit routine
46
47      % Calculate curve-fit
48 -    [fitresult]=fit(x,y,fittype,'robust',robust);
49
50      % Evaluate curve-fit
51 -    poly_eval=feval(fitresult,x);
52
53      % Calculate coefficients for upper and lower confidence bounds
54 -    ci = confint(fitresult,0.95);
55
56      % Evaluate confidence bounds
57 -    PI(:,1) = polyval(ci(1,:),x);
58 -    PI(:,2) = polyval(ci(2,:),x);
```

Figure A3.6. Curve-fitting MATLAB code

Table A3.4. Curve-fitting parameters

Effectivity Index		Conservativeness	
fittype:	'poly2'	fittype:	'poly2'
robust:	'on'	robust:	'off'