PERGAMON

Progress in Aerospace Sciences 38 (2002) 209–272

PROGRESS IN AEROSPACE SCIENCES

www.elsevier.com/locate/paerosci

Verification and validation in computational fluid dynamics

William L. Oberkampf[a,*], Timothy G. Trucano[b]

[a] *Validation and Uncertainty Estimation Department, Sandia National Laboratories, MS 0828, P. O. Box 5800, Albuquerque, NM 87185-0828, USA*

[b] *Optimization and Uncertainty Estimation Department, Sandia National Laboratories, MS 0828, P. O. Box 5800, Albuquerque, NM 87185-0828, USA*

Abstract

Verification and validation (V&V) are the primary means to assess accuracy and reliability in computational simulations. This paper presents an extensive review of the literature in V&V in computational fluid dynamics (CFD), discusses methods and procedures for assessing V&V, and develops a number of extensions to existing ideas. The review of the development of V&V terminology and methodology points out the contributions from members of the operations research, statistics, and CFD communities. Fundamental issues in V&V are addressed, such as code verification versus solution verification, model validation versus solution validation, the distinction between error and uncertainty, conceptual sources of error and uncertainty, and the relationship between validation and prediction. The fundamental strategy of verification is the identification and quantification of errors in the computational model and its solution. In verification activities, the accuracy of a computational solution is primarily measured relative to two types of highly accurate solutions: analytical solutions and highly accurate numerical solutions. Methods for determining the accuracy of numerical solutions are presented and the importance of software testing during verification activities is emphasized. The fundamental strategy of validation is to assess how accurately the computational results compare with the experimental data, with quantified error and uncertainty estimates for both. This strategy employs a hierarchical methodology that segregates and simplifies the physical and coupling phenomena involved in the complex engineering system of interest. A hypersonic cruise missile is used as an example of how this hierarchical structure is formulated. The discussion of validation assessment also encompasses a number of other important topics. A set of guidelines is proposed for designing and conducting validation experiments, supported by an explanation of how validation experiments are different from traditional experiments and testing. A description is given of a relatively new procedure for estimating experimental uncertainty that has proven more effective at estimating random and correlated bias errors in wind-tunnel experiments than traditional methods. Consistent with the authors' contention that nondeterministic simulations are needed in many validation comparisons, a three-step statistical approach is offered for incorporating experimental uncertainties into the computational analysis. The discussion of validation assessment ends with the topic of validation metrics, where two sample problems are used to demonstrate how such metrics should be constructed. In the spirit of advancing the state of the art in V&V, the paper concludes with recommendations of topics for future research and with suggestions for needed changes in the implementation of V&V in production and commercial software.

Contents

*Corresponding author. Tel.: +1-505-844-3799; fax: +1-505-844-4523.
E-mail addresses: wloberk@sandia.gov (W.L. Oberkampf).

0376-0421/02/$ - see front matter
PII: S0376-0421(02)00005-2

1. Introduction

1.1. Background

During the last three or four decades, computer simulations of physical processes have been used in scientific research and in the analysis and design of engineered systems. The systems of interest have been existing or proposed systems that operate at design conditions, off-design conditions, failure-mode conditions, or accident scenarios. The systems of interest have also been natural systems. For example, computer simulations are used for environmental predictions, as in the analysis of surface-water quality and the risk assessment of underground nuclear-waste repositories. These kinds of predictions are beneficial in the development of public policy, in the preparation of safety procedures, and in the determination of legal liability. Thus, because of the impact that modeling and simulation predictions can have, the credibility of the computational results is of great concern to engineering designers and managers, public officials, and those who are affected by the decisions that are based on these predictions.

For engineered systems, terminology such as "virtual prototyping" and "virtual testing" is now being used in engineering development to describe numerical simulation for the design, evaluation, and "testing" of new hardware and even entire systems. This new trend of modeling-and- simulation-based design is primarily

driven by increased competition in many markets, e.g., aircraft, automobiles, propulsion systems, and consumer products, where the need to decrease the time and cost of bringing products to market is intense. This new trend is also driven by the high cost and time that are required for testing laboratory or field components, as well as complete systems. Furthermore, the safety aspects of the product or system represent an important, sometimes dominant element of testing or validating numerical simulations. The potential legal and liability costs of hardware failures can be staggering to a company, the environment, or the public. This consideration is especially critical, given that the reliability, robustness, or safety of some of these computationally based designs are high-consequence systems that cannot ever be tested. Examples are the catastrophic failure of a full-scale containment building for a nuclear power plant, a fire spreading through (or explosive damage to) a high-rise office building, and a nuclear weapon involved in a ground-transportation accident. In computational fluid dynamics (CFD) research simulations, in contrast, an inaccurate or misleading numerical simulation in a conference paper or a journal article has comparatively no impact.

Users and developers of computational simulations today face a critical issue: How should confidence in modeling and simulation be critically assessed? Verification and validation (V&V) of computational simulations are the primary methods for building and quantifying this confidence. Briefly, verification is the assessment of the accuracy of the solution to a computational model by comparison with known solutions. Validation is the assessment of the accuracy of a computational simulation by comparison with experimental data. In verification, the relationship of the simulation to the real world is not an issue. In validation, the relationship between computation and the real world, i.e., experimental data, *is* the issue. Stated differently, verification is primarily a mathematics issue; validation is primarily a physics issue [278].

In the United States, the Defense Modeling and Simulation Office (DMSO) of the Department of Defense has been the leader in the development of fundamental concepts and terminology for V&V [98,100]. Recently, the Accelerated Strategic Computing Initiative (ASCI) of the Department of Energy (DOE) has also taken a strong interest in V&V. The ASCI program is focused on computational physics and computational mechanics, whereas the DMSO has traditionally emphasized high-level systems engineering, such as ballistic missile defense systems, warfare modeling, and simulation-based system acquisition. Of the work conducted by DMSO, Cohen recently observed [73]: "Given the critical importance of model validation…, it is surprising that the constituent parts are not provided in the (DoD) directive concerning… validation. A statistical perspective is almost entirely missing in these directives." We believe this observation properly reflects the state of the art in V&V, not just the directives of DMSO. That is, the state of the art has not developed to the point where one can clearly point out all of the actual methods, procedures, and process steps that *must* be undertaken for V&V. It is our view that the present method of qualitative "graphical validation," i.e., comparison of computational results and experimental data on a graph, is inadequate. This inadequacy especially affects complex engineered systems that heavily rely on computational simulation for understanding their predicted performance, reliability, and safety. We recognize, however, that the complexities of the quantification of V&V are substantial, from both a research perspective and a practical perspective. To indicate the degree of complexity, we suggest referring to quantitative V&V as "validation science."

It is fair to say that computationalists and experimentalists in the field of fluid dynamics have been pioneers in the development of methodology and procedures in validation. However, it is also fair to say that the field of CFD has, in general, proceeded along a path that is largely independent of validation. There are diverse reasons why the CFD community has not perceived a strong need for code V&V, especially validation. A competitive and frequently adversarial relationship (at least in the US) has often existed between computationalists (code users and code writers) and experimentalists, which has led to a lack of cooperation between the two groups. We, on the other hand, view computational simulation and experimental investigations as complementary and synergistic. To those who might say, "Isn't that obvious?" we would answer, "It should be, but they have not always been viewed as complementary." The "line in the sand" was formally drawn in 1975 with the publication of the article "Computers versus Wind Tunnels" [63]. We call attention to this article only to demonstrate, for those who claim it never existed, that a competitive and adversarial relationship has indeed existed in the past, particularly in the US This relationship was, of course, not caused by the quoted article; the article simply brought the competition and conflict to the foreground. In retrospect, the relationship between computationalists and experimentalists is probably understandable because it represents the classic case of a new technology (computational simulation) that is rapidly growing and attracting a great deal of visibility and funding support that had been the domain of the older technology (experimentation).

During the last few years, however, the relationship between computationalists and experimentalists has improved significantly. This change reflects a growing awareness that competition does not best serve the interests of either group [3,38,53,78,106,125,205,207,

212,227,236,237]. Even with this awareness, there are significant challenges in implementing a more cooperative working relationship between the two groups, and in making progress toward a validation science. From the viewpoint of some experimentalists, one of the challenges is overcoming the perceived threat that CFD poses. Validation science requires a close and synergistic working relationship between computationalists and experimentalists, rather than competition. Another significant challenge involves required changes in the perspective of most experimentalists toward validation experiments. We argue that validation experiments are indeed different from traditional experiments, i.e., validation experiments are designed and conducted for the purpose of model validation. For example, there is a critical need for the detailed characterization of the experimental conditions and the uncertainty estimation of the experimental measurements. Similarly, quantitative numerical error estimation by CFD analysts is a must. For complex engineering problems, this requires a posteriori error estimation; not just formal error analyses or a priori error estimation. And finally, we believe validation science will require the incorporation of nondeterministic simulations, i.e., multiple deterministic simulations that reflect uncertainty in experimental parameters, initial conditions, and boundary conditions that exist in the experiments that are used to validate the computational models.

1.2. Outline of the paper

This paper presents an extensive review of the literature in V&V in CFD, discusses methods and procedures for assessing V&V, and develops a number of extensions to existing ideas. Section 2 describes the development of V&V terminology and methodology and points out the various contributions by members of the operations research (OR), statistics, and CFD communities. The development of V&V terminology and methodology is traced back to the OR community, and the accepted terminology in the CFD community is discussed, with differences noted where the terminology differs in the computer science community. The contributions of the CFD community in the development of concepts and procedures for V&V methodology, as well as those in validation experimentation and database construction, are described. Section 2 also summarizes portions of the first engineering standards document published on V&V in CFD [12]. Here we summarize the fundamental approach to verification and validation assessment, five major error sources to be addressed in verification, and the recommended hierarchical, or building-block, validation methodology.

Section 3 discusses the primary methods and procedures of verification assessment. The strategy involves the identification and quantification of errors in the formulation of the discretized model, in the embodiment of the discretized model, i.e., the computer program, and the computation of a numerical solution. The distinction is made between error and uncertainty, code verification and solution verification, and the relationship of verification to software quality engineering. In verification activities, the accuracy of a computational solution is primarily measured relative to two types of highly accurate solutions: analytical solutions and highly accurate numerical solutions. Methods for determining the accuracy of these numerical solutions are presented and the importance of software testing during verification activities is emphasized. A survey of the principal methods used in software quality engineering is also provided.

Section 4 discusses the primary methods and procedures for validation assessment. The strategy of validation is to assess how accurately the computational results compare with the experimental data, with quantified error and uncertainty estimates for both. We begin this section with a discussion of model validation as opposed to "solution validation", the relationship of validation to prediction, four sources of uncertainty and error in validation, and the relationship between validation and calibration. The recommended validation strategy employs a hierarchical methodology that segregates and simplifies the physical and coupling phenomena involved in the complex engineering system of interest. The hierarchical methodology is clearly directed toward validation assessment of models in an engineering analysis environment, not simply a research environment. A hypersonic cruise missile is used as an example of how this hierarchical structure is formulated. The discussion of validation assessment also encompasses a number of other important topics. A set of guidelines is proposed for designing and conducting validation experiments, supported by an explanation of how validation experiments are different from traditional experiments and testing. Next is a description of a relatively new procedure for estimating experimental uncertainty that has proven more effective at estimating random and correlated bias errors in wind-tunnel experiments than traditional methods. Consistent with the authors' contention that nondeterministic simulations are needed in many validation comparisons, a three-step statistical approach is offered for incorporating experimental uncertainties into the computational analysis. Error concepts from hypothesis testing, which is commonly used in the statistical validation of models, are also considered for their possible application in determining and evaluating a metric for comparing the results of a computational simulation with experimental data. The discussion of validation assessment ends with the topic of validation metrics, where two sample problems are used to demonstrate how such metrics should be constructed.

Section 5 makes a number of recommendations for future research topics in computational, mathematical and experimental activities related to V&V. Suggestions are also made for needed improvements in engineering standards activities, the need for implementation of V&V processes in software development, as well as needed changes in experimental activities directed toward validation.

2. Terminology and methodology

2.1. Development of terminology for verification and validation

The issues underlying the V&V of mathematical and computational models of physical processes in nature touch on the very foundations of mathematics and science. Verification will be seen to be rooted in issues of continuum and discrete mathematics and in the accuracy and correctness of complex logical structures (computer codes). Validation is deeply rooted in the question of how formal constructs of nature (mathematical models) can be tested by physical observation. The renowned twentieth-century philosophers of science, Popper [263,264] and Carnap [56], laid the foundation for the present-day concepts of validation. The first technical discipline that began to struggle with the methodology and terminology of verification and validation was the operations research (OR) community, also referred to as systems analysis [27,29,31,54,59,64,71,77,86,93,121, 128,160,181–183,189,190,193,198,217,224–226,248–250, 252,253,293–297,304,345,349]. The fundamental issues of V&V were first debated about 30 to 40 years ago in the OR field. (See [183] for an excellent historical review of the philosophy of science viewpoint of validation. See [30,146,235] for bibliographies in verification and validation.) V&V are specific concepts that are associated with the very general field of modeling and simulation. In the OR activities, the systems analyzed could be extraordinarily complex, e.g., industrial production models, industrial planning models, marketing models, national and world economic models, and military conflict models. These complex models commonly involve a strong coupling of complex physical processes, human behavior, and computer-controlled systems. For such complex systems and processes, fundamental conceptual issues immediately arise with regard to assessing accuracy of the model and the resulting simulations. Indeed, the accuracy of most of these models cannot be validated in any meaningful way.

The high point of much of the early work by the OR community was publication of the first definitions of V&V by the Society for Computer Simulation (SCS) [297]. The published definitions were the following:

Model verification: substantiation that a computerized model represents a conceptual model within specified limits of accuracy.

Model validation: substantiation that a computerized model within its domain of applicability possesses a satisfactory range of accuracy consistent with the intended application of the model.

The SCS definition of verification, although brief, is quite informative. The main implication is that the computerized model, i.e., computer code, must accurately mimic the model that was originally conceptualized. The SCS definition of validation, although instructive, appears somewhat vague. Both definitions, however, contain an important feature: *substantiation*, which is evidence of correctness. Fig. 1 depicts the role of V&V within the phased approach for modeling and simulation adopted by the SCS.

Fig. 1 identifies two types of models: a conceptual model and a computerized model. The conceptual model is composed of all information, mathematical modeling data, and mathematical equations that describe the physical system or process of interest. The conceptual model is produced by analyzing and observing the physical system. In CFD, the conceptual model is dominated by the partial differential equations (PDEs) for conservation of mass, momentum, and energy. In addition, the CFD model includes all of the auxiliary equations, such as turbulence models and chemical reaction models, and all of the initial and boundary conditions of the PDEs. The SCS defined *Qualification* as: Determination of adequacy of the conceptual model to provide an acceptable level of agreement for the domain of intended application. Since we are focusing on V&V, we will not address qualification issues in this paper. The computerized model is an operational computer program that implements a conceptual model.

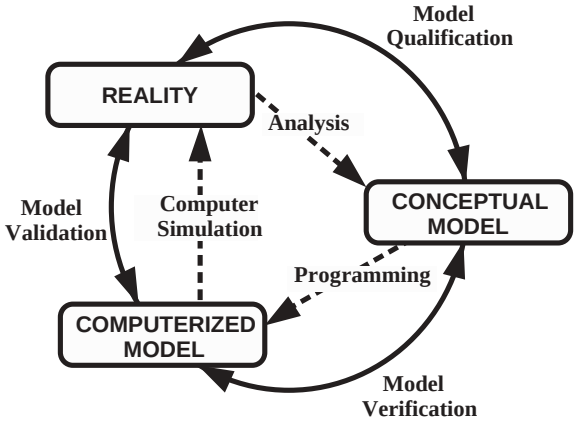

Fig. 1. Phases of modeling and simulation and the role of V&V [297].

Modern terminology refers to the computerized model as the computer model or code. Fig. 1 clearly shows that verification deals with the relationship between the conceptual model and the computerized model and that validation clearly deals with the relationship between the computerized model and reality. These relationships are not always recognized in other definitions of V&V, as will be discussed shortly.

Fundamentally, V&V are tools for assessing the accuracy of the conceptual and computerized models. For much of the OR work, the assessment was so difficult, if not impossible, that V&V became more associated with the issue of credibility, i.e., the quality, capability, or power to elicit belief. In science and engineering, however, quantitative assessment of accuracy, at least for some important physical cases, is mandatory, and in certain situations, assessment can only be conducted using subscale physical models or a subset of the active physical processes. Regardless of the difficulties and constraints, methods must be devised for measuring the accuracy of the model for as many conditions as the model is deemed appropriate. As the complexity of a model increases, its accuracy and range of applicability can become questionable.

During the 1970s, the importance of modeling and simulation dramatically increased as computer-controlled systems started to become widespread in commercial and public systems, particularly automatic flight-control systems for aircraft. At about the same time, the first commercial nuclear-power reactors were being built, and issues of public safety in normal operational environments and accident environments were being critically questioned. In response to this interest, the Institute of Electrical and Electronics Engineers (IEEE) defined verification as follows [167,168]:

Verification: the process of evaluating the products of a software development phase to provide assurance that they meet the requirements defined for them by the previous phase.

This IEEE definition is quite general and it is referential; that is, the value of the definition is related to the definition of "requirements defined for them by the previous phase." Because those requirements are not stated in the definition, the definition does not contribute much to the intuitive understanding of verification or to the development of specific methods for verification. While the definition clearly includes a requirement for the consistency of products (e.g., computer programming) from one phase to another, the definition does not contain a specific requirement for correctness or accuracy.

At the same time, IEEE defined validation as follows [167,168]:

Validation: the process of testing a computer program and evaluating the results to ensure compliance with specific requirements.

The IEEE definitions emphasize that both V&V are processes, that is, ongoing activities. The definition of validation is also referential because of the phrase "compliance with specific requirements." Because specific requirements are not defined (to make the definition as generally applicable as possible), the definition of validation is not particularly useful by itself. The substance of the meaning must be provided in the specification of additional information. Essentially the same definitions for V&V have been adopted by the American Nuclear Society (ANS) [19] and the International Organization for Standardization (ISO) [170].

One may ask why the IEEE definitions are included since they seem to provide less understanding than the earlier definitions of the SCS. First, these definitions provide a distinctly different perspective toward the entire issue of V&V than what is needed in CFD and computational mechanics. This perspective asserts that because of the extreme variety of requirements for modeling and simulation, the requirements should be defined in a separate document for each application, not in the definitions of V&V. Second, the IEEE definitions are the more prevalent definitions used in engineering, and one must be aware of the potential confusion when other definitions are used. The IEEE definitions are dominant because of the worldwide influence of this organization. It should be noted that the IEEE definitions are also used by the computer science community and the software quality assurance community. Given that members of these two communities often work together with the computational mechanics community, we expect there to be long-term ambiguity and confusion in the terminology.

The Defense Modeling and Simulation Organization (DMSO) of the US Department of Defense (DoD) has also played a major role in attempting to standardize the definitions of V&V. For this standardization effort, the DMSO obtained the expertise of researchers in the fields of OR, operational testing of combined hardware and software systems, man-in-the-loop training simulators, and warfare simulation. Recently, the DoD published definitions of V&V that are clear, concise, and directly useful by themselves [99,100]. Also during the early 1990s, the CFD Committee on Standards of the American Institute of Aeronautics and Astronautics (AIAA) was also discussing and debating definitions of V&V. Because the AIAA committee essentially adopted the DMSO definitions, they will be discussed together.

The DMSO definition of verification, although directly useful, does not make it clear that the accuracy of the numerical solution to the conceptual model should be included in the definition. The reason for this lack of clarity is that the numerical solution of PDEs was not a critical factor in DMSO's perspective of what verification is intended to accomplish. The AIAA,

however, was primarily interested in the accuracy of the numerical solution—a concern that is common to essentially all of the fields in computational sciences and engineering, such as computational mechanics, structural dynamics, and computational heat transfer. Consequently, the AIAA slightly modified the DMSO definition for verification and adopted verbatim the DMSO definition of validation. The AIAA definitions are given as follows [12]:

Verification: the process of determining that a model implementation accurately represents the developer's conceptual description of the model and the solution to the model.

Validation: the process of determining the degree to which a model is an accurate representation of the real world from the perspective of the intended uses of the model.

By comparing the AIAA definitions with the definitions that were developed by the SCS and represented in Fig. 1, one finds that the definitions from the two organizations are fundamentally identical—though the AIAA definitions are clearer and more intuitively understandable, particularly the definition for validation. The AIAA definition for validation directly addresses the issue of fidelity of the computational model to the real world.

There are some important implications and subtleties in the definitions of V&V that should be addressed. The first significant feature is that both V&V are "process[es] of determining." That is, they are ongoing activities that do not have a clearly defined completion point. Completion or sufficiency is usually determined by practical issues such as budgetary constraints and intended uses of the model. The definitions include the ongoing nature of the process because of an unavoidable but distressing fact: the veracity, correctness, and accuracy of a computational model cannot be demonstrated for all possible conditions and applications, except for trivial models. Trivial models are clearly not of interest. All-encompassing proofs of correctness, such as those developed in mathematical analysis and logic, do not exist in complex modeling and simulation. Indeed, one cannot prove that complex computer codes have no errors. Likewise, models of physics cannot be proven correct, they can only be disproved. Thus, V&V activities can only assess the correctness or accuracy of the specific cases tested.

The emphasis on "accuracy" is the second feature that is common in the definitions of V&V. This feature assumes that a measure of correctness can be determined. In verification activities, accuracy is generally measured in relation to benchmark solutions of simplified model problems. Benchmark solutions refer to either analytical solutions or highly accurate numerical solutions. In validation activities, accuracy is measured in relation to experimental data, i.e., our best indication of reality. However, benchmark solutions and experimental data also have shortcomings. For example, benchmark solutions are extremely limited in the complexity of flow physics and geometry; and all experimental data have random (statistical) and bias (systematic) errors that may cause the measurements to be less accurate than the corresponding computational results in some situations. These issues are discussed in more detail later in this paper.

Effectively, verification provides evidence (substantiation) that the conceptual (continuum mathematics[1]) model is solved correctly by the discrete mathematics computer code. Verification does not address whether the conceptual model has any relationship to the real world. Validation, on the other hand, provides evidence (substantiation) for how accurately the computational model simulates reality. This perspective implies that the model is solved correctly, or verified. However, multiple errors or inaccuracies can cancel one another and give the appearance of a validated solution. Verification, thus, is the first step of the validation process and, while not simple, is much less involved than the more comprehensive nature of validation. Validation addresses the question of the fidelity of the model to specific conditions of the real world. The terms "evidence" and "fidelity" both imply the concept of "estimation of error," not simply "yes" or "no" answers.

2.2. Contributions in fluid dynamics

In science and engineering, CFD was one of the first fields to seriously begin developing concepts and procedures for V&V methodology. Our review of the literature has identified a number of authors who have contributed to the verification of CFD solutions [11, 28,33,34,39–42,45,47,49,57,60,65,74,87–89,92,97,110, 116, 117, 120, 127, 129–131,137–139,141,142,144,147,165, 171, 177, 209, 218, 219, 221, 228, 229,273,274,277,278,281, 298, 306–309, 318, 325, 327,329,333–337,340,347,350,351, 353]. Most of these authors have contributed highly accurate numerical solutions, while some have contributed analytical solutions useful for verification. In addition, several of these authors have contributed to the numerical methods needed in verification activities, e.g., development of procedures using Richardson extrapolation. Note, however, that many of these authors, especially those in the early years, do not refer to verification or may even refer to their work as "validation" benchmarks. This practice simply reflects

[1] When we refer to "continuum mathematics" we are not referring to the physics being modeled by the mathematics. For example, the equations for noncontinuum fluid dynamics are commonly expressed with continuum mathematics. Additionally, our discussion of V&V does not restrict the mathematics to be continuum in any substantive way.

the past, and even present, confusion and ambiguity in the terminology. Several textbooks, including [151,176, 223,256,341,343], also contain a number of analytical solutions that are useful for verification of CFD codes.

Work in validation methodology and validation experiments has also been conducted by a large number of researchers through the years. Examples of this published work can be found in [2–4,8–10,32,37,38, 45, 53,55,58,76,79–81,90,101,102,106,123,135,136,152,163, 166, 174, 201, 202, 204–208, 211, 213, 214,227,234,236–238, 240, 241, 246, 254, 265, 269, 272, 274, 276, 278,279,302,310–313,317,324,326,330,331,338,342]. Much of this work has focused on issues such as fundamental methodology and terminology in V&V, development of the concepts and procedures for validation experiments, confidence in predictions based on validated simulations, and methods of incorporating validation into the engineering design process. Some of this work has dealt with experiments that were specifically designed as validation experiments. (This topic is discussed in detail in Section 4.) Essentially all of this early work dealt with CFD for aircraft and reentry vehicle aerodynamics, gas turbine engines, and turbopumps.

In parallel with the aerospace activities, there have been significant efforts in V&V methodology in the field of environmental quality modeling—most notably, efforts to model the quality of surface and ground water and to assess the safety of underground radioactive-waste repositories [35,85,96,178,196,251, 282,288,299,305,323,328,352]. This water-quality work is significant for two reasons. First, it addresses validation for complex processes in the physical sciences where validation of models is extremely difficult, if not impossible. One reason for this difficulty is extremely limited knowledge of underground transport and material properties. For such situations, one must deal with calibration or parameter estimation in models. Second, because of the limited knowledge, the environmental-modeling field has adopted statistical methods of calibration and validation assessment.

The typical validation procedure in CFD, as well as other fields, involves graphical comparison of computational results and experimental data. If the computational results "generally agree" with the experimental data, the computational results are declared "validated." Comparison of computational results and experimental data on a graph, however, is only incrementally better than a qualitative comparison. With a graphical comparison, one does not commonly see quantification of the numerical error or quantification of computational uncertainties due to missing initial conditions, boundary conditions, or modeling parameters. Also, an estimate of experimental uncertainty is not typically quoted, and in most cases it is not even available. In the event that computational uncertainty due to missing data or experimental uncertainty was available, statistical comparisons would be required. A graphical comparison also gives little quantitative indication of how the agreement varies over the range of the independent variable, e.g., space or time, or the parameter of interest, e.g., Reynolds number or a geometric parameter. An important issue concerns how comparisons of computational results and experimental data could be better quantified. We suggest that validation quantification should be considered as the evaluation of a metric, or a variety of appropriate metrics, for measuring the consistency of a given computational model with respect to experimental measurements. A metric would quantify both errors and uncertainties in the comparison of computational results and experimental data. Only a few researchers have pursued this topic [35,76,103,128,148,157, 181,192,194–196,246]. And of these researchers, only two [76,246] are in the field of fluids-engineering systems.

An additional important thread in the increasing importance and visibility of validation has been the construction and dissemination of experimental databases. Some of the important work in this area was performed by the NATO Advisory Group for Aerospace Research and Development (AGARD) and several independent researchers [1,5–10,38,94,95, 134,162,163,207,300–302]. Participants in the Fluid Dynamics Panel of AGARD recognized very early the importance of validation databases in the development and maturing of CFD for use in engineering applications. Although many of the defining characteristics of true validation experiments were lacking in the early work, the work of the Fluid Dynamics Panel helped develop those characteristics. V&V databases for CFD are now appearing on the World Wide Web, with some of the most important of these described in [109,118,220,222,230]. The Web provides extraordinarily easy access to these databases not only within a country but also around the globe. However, most of these databases have restricted access to part or all of the database. Even though some initiatives have been started for constructing verification and validation databases, we believe it is fair to characterize most of the present effort as ad hoc and duplicative.

An examination of the literature from the diverse disciplines of OR, CFD (primarily for aerospace sciences), and environmental sciences clearly shows that each discipline developed concepts and procedures essentially independently. Each of these fields has emphasized different aspects of V&V, such as terminology and philosophy, numerical error estimation, experimental methods and uncertainty estimation, benefits of V&V in the engineering design environment, and uses of V&V in environmental risk assessment using modeling and simulation. This paper attempts to inform the aerospace-sciences CFD community about the productive and beneficial work performed by the OR

community and other CFD communities like environmental sciences.

2.3. Methodology for verification

In 1992, the AIAA Computational Fluid Dynamics Committee on Standards began a project to formulate and standardize the basic terminology and methodology in the V&V of CFD simulations. The committee is composed of representatives from academia, industry, and government, with representation from the US, Canada, Japan, Belgium, Australia, and Italy. After six years of discussion and debate, the committee's project culminated in the publication of *Guide for the Verification and Validation of Computational Fluid Dynamics Simulations* [12], referred to herein as the "AIAA Guide." The AIAA Guide defines a number of key terms, discusses fundamental concepts, and specifies general procedures for conducting V&V in CFD.

AIAA standards documents are segregated into three levels that reflect the state of the art: guides, recommended practices, and standards. The AIAA Guide is at the first level, denoting the early stage of development of concepts and procedures in V&V. It is also the first standards-like document to be published by any engineering society on the topic of V&V. The American Society of Mechanical Engineers (ASME) has recently formed a new standards committee and is developing a similar document in the field of solid mechanics [20]. In this section, we briefly review portions of the AIAA Guide that deal with fundamental V&V methodology.

Verification is the process of determining that a model implementation accurately represents the developer's conceptual description of the model and the solution to the model. The fundamental strategy of verification is the identification, quantification, and reduction of errors in the computational model and its solution. To quantify numerical solution error, a highly accurate, reliable fiducial (benchmark) must be available. Highly accurate solutions refer to either analytical solutions or highly accurate numerical solutions. Highly accurate solutions, unfortunately, are only available for simplified model problems. Verification, thus, provides evidence (substantiation) that the conceptual (continuum mathematics) model is solved correctly by the discrete mathematics embodied in the computer code. The conceptual model does not require any relationship to the real world. As Roache [278] lucidly points out, verification is a mathematics and computer science issue; not a physics issue. Validation is a physical sciences and mathematics issue. Fig. 2 depicts the verification process of comparing the numerical solution from the code in question with various types of highly accurate solutions.

Verification activities are primarily performed early in the development cycle of a computational code. However, these activities must be repeated when the code is subsequently modified or enhanced. Although the required accuracy of the numerical solutions that are obtained during validation activities depends on the problem to be solved and the intended uses of the code, the accuracy requirements in verification activities are generally more stringent than the accuracy requirements in validation activities. Note that the recommended methodology presented here applies to finite difference, finite volume, finite element, and boundary element discretization procedures. An extensive description of verification methodology and procedures is given in [278].

Given a numerical procedure that is stable, consistent, and robust, the five major sources of errors in CFD solutions are insufficient spatial discretization convergence, insufficient temporal discretization convergence, insufficient convergence of an iterative procedure, computer round-off, and computer programming errors. The emphasis in verification is the identification and quantification of these errors, as well as the demonstration of the stability, consistency, and robustness of the numerical scheme. Stated differently, an analytical or formal error analysis is inadequate in the verification process; verification relies on demonstration. Error bounds, such as global or local error norms must be computed in order to quantitatively assess the test of agreement shown in Fig. 2. Upon examining the CFD literature as a whole, it is our view that verification testing of computer codes is severely inadequate. The predominant view in the field of CFD, as in scientific software in general, is that little verification testing is necessary prior to utilization. We believe this view is indicative of the relative immaturity of the field of CFD, that is, much of the literature is of a research nature or concentrates on numerical issues such as solution algorithms, grid construction, or visualization. When CFD is applied to a real-world engineering application, such as design or risk assessment, there is typically an abundance of experimental data or experience with the application. For confidence in CFD to improve,

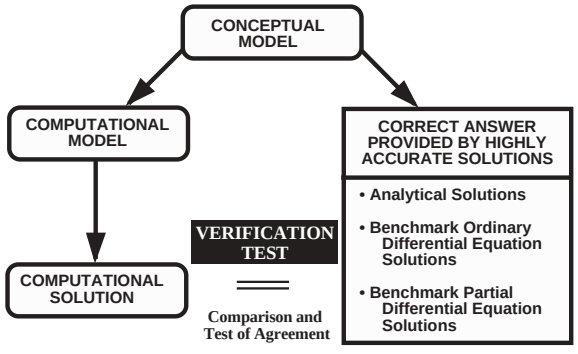

Fig. 2. Verification process [12].

especially for engineering applications where there is little experimental data or experience, we contend that additional effort should be expended on verification testing.

The first three error sources listed above (spatial discretization error, temporal discretization error, and iterative error) are considered to be within the traditional realm of CFD, and there is extensive literature, some of it referenced in Section 2.2, dealing with each of these topics. The fourth error source, computer round-off, is rarely dealt with in CFD. Collectively, these four topics in verification could be referred to as solution verification or solution accuracy assessment. The fifth source, programming errors, is generally considered to be in the realm of computer science or software engineering. Programming errors, which can occur in input data files, source code programming, output data files, compilers, and operating systems, are addressed using methods and tools in software quality assurance, also referred to as software quality engineering (SQE) [186]. The identification of programming errors could be referred to as code verification, as opposed to solution verification [278,279]. Although the CFD community generally has given little attention to SQE, we believe that more effort should be devoted to this topic. Section 3 discusses issues in SQE, but the primary emphasis in that section is on estimation of error from the first three sources.

2.4. Methodology for validation

As discussed in the AIAA Guide, validation is the process of determining the degree to which a model is an accurate representation of the real world from the perspective of the intended uses of the model [12]. Validation deals with the assessment of the comparison between sufficiently accurate computational results and the experimental data. Validation does not specifically address how the computational model can be changed to improve the agreement between the computational results and the experimental data, nor does it specifically address the inference of the model's accuracy for cases different from the validation comparison. The fundamental strategy of validation involves identification and quantification of the error and uncertainty in the conceptual and computational models, quantification of the numerical error in the computational solution, estimation of the experimental uncertainty, and finally, comparison between the computational results and the experimental data. That is, accuracy is measured in relation to experimental data, our best measure of reality. This strategy *does not* assume that the experimental measurements are more accurate than the computational results. The strategy only asserts that experimental measurements are the most faithful reflections of reality for the purposes of validation. Validation requires that the estimation process for error and uncertainty must occur on both sides of the coin: mathematical physics and experiment. Fig. 3 depicts the validation process of comparing the computational results of the modeling and simulation process with experimental data from various sources.

Because of the infeasibility and impracticality of conducting true validation experiments on most complex systems, the recommended method is to use a building-block approach [12,79,201,207,310,311]. This approach divides the complex engineering system of interest into three, or more, progressively simpler tiers: subsystem cases, benchmark cases, and unit problems. (Note that in the AIAA Guide, the building-block tiers are referred to as phases.) The strategy in the tiered approach is to assess how accurately the computational results compare with the experimental data (with quantified uncertainty estimates) at multiple degrees of physics coupling and geometric complexity (see Fig. 4).

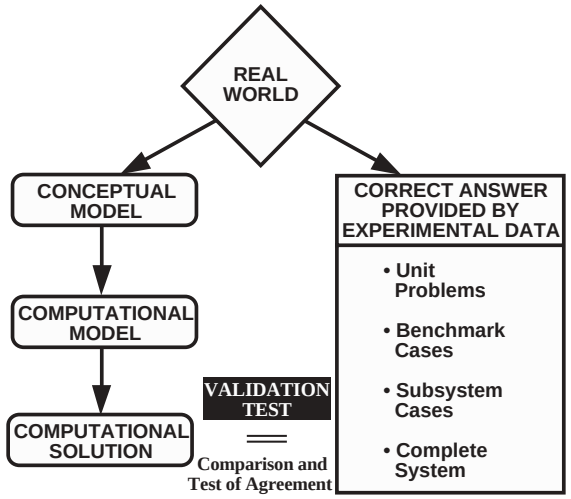

Fig. 3. Validation process [12].

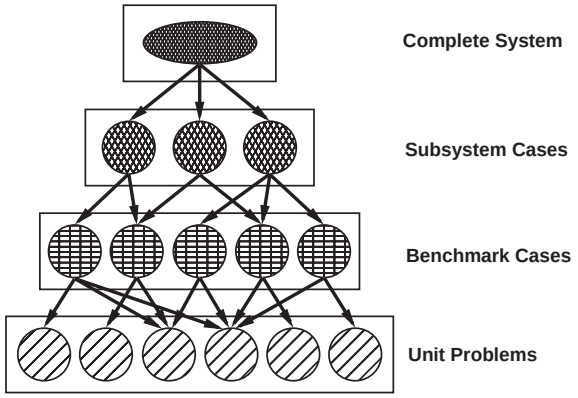

Fig. 4. Validation tiers [12].

The approach is clearly constructive in that it (1) recognizes that there is a hierarchy of complexity in systems and simulations and (2) recognizes that the quantity and accuracy of information that is obtained from experiments varies radically over the range of tiers. It should also be noted that additional building-block tiers beyond the four that are discussed here could be defined, but additional tiers would not fundamentally alter the recommended methodology.

The complete system consists of the actual engineering hardware for which a reliable computational tool is needed. Thus, by definition, all the geometric and physics effects occur simultaneously. For typical complex engineering systems (e.g., a gas turbine engine), multidisciplinary, coupled physical phenomena occur together. Data are measured on the engineering hardware under realistic operating conditions. The quantity and quality of these measurements, however, are essentially always very limited. It is difficult, and sometimes impossible, for complex systems to quantify most of the test conditions required for computational modeling, e.g., various fluid flow-rates, thermophysical properties of the multiple fluids, and coupled time-dependent boundary conditions. Not only are many required modeling parameters unmeasured, no experimental uncertainty analysis is generally conducted.

Subsystem cases represent the first decomposition of the actual hardware into simplified systems or components. Each of the subsystems or components is composed of actual hardware from the complete system. Subsystem cases usually exhibit three or more types of physics that are coupled. Examples of types of physics are fluid dynamics, structural dynamics, solid dynamics, chemical reactions, and acoustics. The physical processes of the complete system are partially represented by the subsystem cases, but the degree of coupling between various physical phenomena in the subsystem cases is typically reduced. For example, there is normally reduced coupling between subsystems as compared to the complete system. The geometric features are restricted to the subsystem and its attachment, or simplified connection, to the complete system. Although the quality and quantity of the test data are usually significantly better for subsystem cases than for the complete system, there are still limited test data for subsystem cases. Some of the necessary modeling data, initial conditions, and boundary conditions are measured, particularly the most important data.

Experimental data from complete systems and data from subsystem tests are always specific to existing hardware and are available mainly through large-scale test programs. Existing data from these tests have traditionally focused on issues such as the functionality, performance, safety, or reliability of systems or subsystems. For large-scale tests, there is often competition between alternative system, subsystem, or component designs. If the competition is due to outside organizations or suppliers of hardware, then the ability to obtain complete and unbiased validation information becomes even more difficult. Such tests generally provide only data that are related to engineering parameters of clear design interest, system functionality, and high-level system performance measures. The obtained data typically have large uncertainty bands, or no attempt has been made to estimate uncertainty. The test programs typically require expensive ground-test facilities, or the programs are conducted in uncontrolled, hostile, or unmeasured environments. Commonly, the test programs are conducted on a rigid schedule and with a tightly constrained budget. Consequently, it is not possible to obtain the complete set of physical modeling parameters (e.g., thermochemical fluid and material properties), initial conditions, and boundary conditions that are required for quantitative validation assessment. Also, there are certain situations where it is not possible to conduct a validation experiment of the complete system. Such situations could involve public-safety or environmental-safety hazards, unattainable experimental-testing requirements, or international treaty restrictions.

Benchmark cases represent the next level of decomposition of the complete system. For these cases, special hardware is fabricated to represent the main features of each subsystem. By special hardware, we mean hardware that is specially fabricated with simplified properties or materials. For example, benchmark hardware is normally not functional hardware nor is it fabricated with the same materials as actual subsystems or components. For benchmark cases, typically only two or three types of coupled flow physics are considered. For example, in fluid dynamics one could have turbulence, combustion, and two-phase flow, but one would eliminate any structural dynamics coupling that might exist at the subsystem level. The benchmark cases are normally simpler geometrically than those cases at the subsystem level. The only geometric features that are retained from the subsystem level are those that are critical to the types of physics that are considered at the benchmark level. Most of the experimental data that are obtained in benchmark cases have associated estimates of measurement uncertainties. Most of the required modeling data, initial conditions, and boundary conditions are measured, but some of the less important experimental data have not been measured. The experimental data, both code input data and system response data, are usually documented with moderate detail. Examples of important experimental data that are documented include detailed inspection of all hardware, specific characterization of materials and fluids used in the experiment, and detailed measurement of environmental conditions that were produced by the experimental apparatus or testing equipment.

Unit problems represent the total decomposition of the complete system. At this level, high-precision, special-purpose hardware is fabricated and inspected, but this hardware rarely resembles the hardware of the subsystem from which it originated. One element of complex physics is allowed to occur in each of the unit problems that are examined. The purpose of these problems is to isolate elements of complex physics so that critical evaluations of mathematical models or submodels can be evaluated. For example, unit problems could individually involve turbulence, laminar flow combustion, or laminar gas/liquid droplet flows. Unit problems are characterized by very simple geometries. The geometry features are commonly two-dimensional, but they can be very simple three-dimensional geometries with important geometric symmetry features. Highly instrumented, highly accurate experimental data are obtained from unit problems, and an extensive uncertainty analysis of the experimental data is prepared. If possible, experiments on unit problems are repeated at separate facilities to ensure that bias (systematic) errors in the experimental data are identified. For unit problems, all of the important code input data, initial conditions, and boundary conditions are accurately measured. These types of experiments are commonly conducted in universities or in research laboratories.

Experimental data from benchmark cases and unit problems should be of the quantity and quality that are required in true validation experiments. If, however, significant parameters that are needed for the CFD simulation of benchmark cases and unit-problem experiments were not measured, then the analyst must assume these quantities. For example, suppose for benchmark cases and unit problems that careful dimensional measurements and specific material properties of the hardware were not made. In this case, the computational analyst typically assumes reasonable or plausible values for the missing data, possibly from an engineering handbook. An alternative technique, although rarely done in CFD, is for the analyst to assume probability distributions with specified means and variances of the unmeasured parameters. Multiple computations are then performed using these assumptions, and likelihoods are computed for output quantities used to compare with experimental data. In existing or older published experimental data for benchmark cases and unit problems, it is common that a significant number of parameters are missing from the description of the experiment. Experiments in the past were typically conducted for the purpose of improving the physical understanding of specific phenomena or for determining parameters in models, rather than for the validation of CFD models. That is, these experiments were used inductively to construct mathematical models of physical phenomena, rather than deductively to evaluate the validity of models.

3. Verification assessment

3.1. Introduction

The verification assessment activities discussed in this paper apply to finite-difference, finite-volume, and finite-element discretization procedures. Roache has extensively treated the subject of verification in his book [278], and a detailed description of verification methodology and procedures can be found there. It is not the purpose of our current treatment to completely review material that is otherwise available in that reference and others. Instead, we desire to summarize key features of verification assessment that serve to emphasize its role as an important partner with validation assessment. We review selected topics in verification assessment for CFD that illustrate most clearly the importance of such assessment.

3.2. Fundamentals of verification

3.2.1. Definitions and general principles

This section discusses some fundamentals of verification that are required to place the remainder of this section in proper perspective. We emphasize the proper and necessary role of verification, as well as its relation to validation. We give a discussion of error estimation and error quantification in verifying calculations, as well as the distinction between verifying codes and verifying calculations. Proper focus on error has impact on both of these assessment activities. In fact, verification assessment may be viewed as the process that minimizes our belief that there are errors in the code, as well as in particular calculations. Verification assessment can be considered as an optimization problem that is certainly constrained by available resources and needs, as well as by intended applications of the code. We also introduce the appropriate role of *software engineering* (SE), also called *software quality engineering* (SQE), in verification assessment.

Verification as defined in Section 2 is an equal partner to validation in the overall verification and validation strategy for a code and its applications. Validation depends on solution accuracy as well as on experimental accuracy. For example, computational error that arises from failure to adequately converge a calculation contributes to the discrepancy or apparent agreement between that calculation and the results of an experiment with which the calculation is compared when validation is performed. If severe enough, such a strictly computational error could dominate this discrepancy. For example, it raises the real possibility in complex problems of confusing a coding error with a mathematical modeling error. The goal should be to distinguish errors in mathematical modeling accuracy as clearly as possible from other errors.

The study of solution error is fundamentally empirical. Achieving the goal of *rigorously* demonstrating that solution error is small for all feasible applications of a CFD code is essentially impossible for complex codes. However, such a goal may be feasible for particular calculations using those codes. In this sense, verification assessment is quite similar to validation assessment. We aim to understand what the computational error is for given calculations or types of calculations, not for codes in general. In many cases, this error may be understood to be small enough to justify the beginning of validation assessment, or continuation of validation assessment in different directions. We can expect that over time we will accumulate increasing amounts of empirical evidence that the error is indeed small enough for important classes of computations, rather than simply for single calculations. Thus we can hope to generalize the understanding we acquire from the study of specific calculations. It is unlikely, however, that rigorous generalization beyond this approach will be possible for the foreseeable future.

Assessment of the accuracy of code solutions, given the underlying equations, is the basic objective of verification assessment. Developing confidence that the code solution is accurate for problems other than verification tests is a key goal of this effort. Therefore, an obvious requirement is to perform the required verification activities in a way that maximizes confidence in the accuracy of new calculations. We will call this the *confidence optimization problem*.

In verification assessment, the goal of achieving a predictive understanding of the numerical accuracy of future application of codes is strongly dependent upon a detailed understanding of the numerical accuracy of past applications of the code. There are several major themes of verification assessment that enable the achievement of this goal. These include (1) classical concepts of convergence of discretizations of partial differential equations (PDEs), especially a posteriori error estimates; (2) the formalization of testing in performing verification assessment; (3) the role of benchmarks as accuracy quantification standards; and (4) the impact of software quality engineering (SQE). These themes are discussed below.

The verification process traditionally rests upon comparing computational solutions to the "correct answer," which is provided by "highly accurate solutions" for a set of well-chosen test problems. As a result, this is a test-dominated strategy for solving the problem of optimizing confidence. The resulting confidence is a specific product of the knowledge that computational error is acceptably small on such a set of tests.

The "correct answer" can only be known in a relatively small number of isolated cases. These cases therefore assume a very important role in verification and are usually carefully formalized in test plans for verification assessment of the code. However, because the elements that constitute such test plans are sparse, there should be some level of concern in the CFD community when these plans are the dominant content of the verification assessment activities. Elements of verification assessment that transcend testing, such as SQE practices [113,215,260,292], are also important. Such elements provide supplemental evidence that the CFD code, as written, has a minimal number of errors.

Verification activities are primarily initiated early in the development cycle of a code. This timing further emphasizes the connection between testing alone and more formal SQE activities that are undertaken as part of the software development and maintenance process. One basic problem that underlies the distinction between testing alone and other elements of the verification process is ultimately how constrained resources can be used to achieve the highest confidence in performance of the code. For simplification of our presentation, however, unless it is obvious from our context, we will use the default understanding that "verification" means "testing" in the remainder of this section.

Because of the code-centric nature of many verification activities, the common language used in discussing verification often refers to *code verification*. What does this concept really mean? In our view, to rigorously verify a code requires rigorous proof that the computational implementation accurately represents the conceptual model and its solution. This, in turn, requires *proof* that the algorithms implemented in the code correctly approximate the underlying PDEs, along with the stated initial and boundary conditions. In addition, it must also be *proven* that the algorithms converge to the correct solutions of these equations in all circumstances under which the code will be applied. It is unlikely that such proofs will ever exist for CFD codes. The inability to provide proof of code verification in this regard is quite similar to the problems posed by validation. Verification, in an operational sense, then becomes the absence of proof that the code is incorrect. While it is possible to prove that a code is functioning incorrectly, it is effectively impossible to prove the opposite. Single examples suffice to demonstrate incorrect functioning, which is also a reason why testing occupies such a large part of the validation assessment effort.

Defining verification as the absence of proof that the code is wrong is unappealing from several perspectives. For example, that state of affairs could result from complete inaction on the part of the code developers or their user community. An activist definition that still captures the philosophical gist of the above discussion is preferable and has been stressed by Peercy [261]. In this definition, verification of a code is equivalent to the development of a legal case. Thus verification assessment consists of accumulating evidence substantiating

that the code does not have algorithmic or programming errors, that the code functions properly on the chosen hardware, and so on. This evidence needs to be documented, accessible, referenceable, and repeatable. The accumulation of such evidence also serves to reduce the regimes of operation of the code where one might possibly find such errors. Confidence in the verification status of the code then results from the accumulation of a sufficient mass of evidence.

The present view of code verification as a continuing, ongoing process, akin to accumulating evidence for a legal case, is not universally accepted. In an alternative view [278], code verification is not ongoing but reaches a termination, more akin to proving a theorem. Obviously, the termination can only be applied to a fixed code; if the code is modified, it is a new code (even if the name of the code remains) and the new code must be re-verified. Also, all plausible *nonindependent* combinations of input options must be exercised so that every line of code is executed in order to claim that the entire code is verified; otherwise, the verification can be claimed only for the subset of options exercised. The ongoing code exercise by multiple users still is useful, in an evidentiary sense (and in user training), but is referred to as *confirmation* rather than code verification. In this alternative view of verification, it is argued that contractual and/or regulatory requirements for delivery or use of a "verified code" can be more easily met, and superficial exercises are less likely to be claimed as partial verification. Ongoing use and exercise can possibly uncover mistakes missed in the code verification process, just as a theorem might turn out to have a faulty proof or to have been misinterpreted, but (in this view) the code verification can be completed, at least in principle. Verification of individual calculations, and certainly validations, are still viewed as ongoing processes, of course.

The primary approach to proving that implemented code is a rigorously correct representation of the underlying conceptual model is through the use of *formal methods*. A great deal of effort has recently been devoted to the development and application of these methods [51,258,287]. The actual goal of formal methods—rigorous "proof" that a system of software is correctly implemented—remains controversial in our opinion. The application of these methods is also complicated by disputes over issues of cost, appropriateness, utility, and impact [50].

Formal methods have certainly not been applied to software systems characteristic of those of interest in CFD, namely those systems in which floating point arithmetic is dominant and an effectively infinite variability in software application is the norm. The utility of these methods to CFD codes, even if resource constraints were not issues, is an unsolved problem. This fact has led to interest in the application of formal methods to more restricted problems that still may be of significant interest in CFD software. For example, there is current interest in applying formal methods to aid in verification of the mathematical formalism of the conceptual model that underlies a CFD code rather than in the full details of the software implementation [72,188].

3.2.2. Developing the case for code verification

How should evidence supporting confidence in code verification be assembled in greater detail? It is clear that accumulating evidence for code verification is a multifold problem. For example, at the very beginning of software implementation of algorithms, any numerical analysis of the algorithms implemented in the code becomes a fundamental check for code verification. Given the simplifying assumptions that underlie most effective numerical analysis (such as linearity of the PDEs), failure of the algorithms to obey the constraints imposed by numerical analysis can only reduce our confidence in the verification status of the code as a whole. As verification assessment proceeds, we are necessarily forced to relax the simplifying assumptions that facilitated numerical analysis. Evidence therefore becomes increasingly characterized by computational experiments and the resulting empirical evidence of code performance.

The scientific community places great emphasis on the evidence developed by executing test problems with the code that is undergoing verification. In fact, testing of this type (including the "free" testing that occurs by allowing users to apply the code) is generally considered to be the most important source of verification evidence for CFD and engineering codes [18,278]. As is emphasized in Section 3.4, this evidence would be greatly strengthened by the formation and application of agreed-upon test suites and of standards for comparing results of calculations with the benchmarks established by the tests, as well as by community agreement on the levels of accuracy that are required to provide evidence that the code has correctly solved the test problem.

Gathering evidence from the user community on the performance of a CFD code contributes to verification assessment. We distinguish this information from formal planned testing because user evidence is not typically accumulated by executing a rational test plan (such as would enter into the design of a test suite for the code). The accumulation of user evidence has some characteristics similar to what is called *random testing* in the software testing literature [175], a topic we also discuss briefly in Section 3.4. Even though user testing is ad hoc compared to formal planned testing, it is the type of testing that is typically associated with verification assessment of CFD software. User testing is known to be incredibly effective at uncovering errors in codes and typically contributes to code verification by uncovering errors that are subsequently fixed, although this may be

accompanied by a corresponding loss of confidence in the code by the user. User testing is also a major cost that can conveniently be hidden from the overall cost and effort associated with code verification assessment centered on the software developers. Both formal controlled testing and "random" user testing are important in CFD verification assessment. But assembling the evidence that results from both of these activities requires care.

The growing impact associated with software failure, the increased size and cost of code projects, and the need for interaction of large numbers of software developers greatly leverage the involvement of and reliance upon formal SE (software engineering) activities in the development of CFD software. Resistance to the application of formal SE in CFD is created by the degree of formality, constraints, and seeming costs of SE, as well as by psychological reasons (science versus product development, for example). As mentioned by Roache [278], SE issues are downright unattractive to small CFD projects whose prime purpose is to develop software for exploration and scientific insight, rather than to minimize the risk of software failure at seemingly great cost.

Significantly, as the goals and expectations of CFD evolve to have impact that goes far beyond scientific exploration and insight, the consequence of software failure greatly magnifies. In particular, inaccurate answers rather than obvious code failures are especially dreaded in high-consequence computing because it may be very difficult to determine that a calculation is inaccurate. SE methodologies provide some additional means of addressing such a problem. But does the cost justify intensive application of such methodologies for a given CFD code project? Part of the confidence optimization problem for verification is the difficult question of how to measure the consequence of CFD code failure. Such a measure is important in clearly understanding when the application of formal SE methodologies provides unquestioned value in code verification assessment.

3.2.3. Error and the verification of calculations

As emphasized by Roache [278], we distinguish the activities of code verification as outlined above from those of calculation verification. Verification of calculations is centered on the accumulation of evidence that a specific calculation is correct and accurate. Even this may well be asking too much, depending on the complexity of a given calculation. For example, calculation verification requires confirmation of grid convergence, which may not be practically attainable. Evidence that the converged calculation is correct may be available if test problems similar to the calculation in question have been calculated with well-understood and well-characterized high accuracy. But any sufficiently new calculation that is not similar to existing test problems will not be subject to this logic and the question of correctness will have to be addressed in some other way. Even under the simplifying assumption of verifying specific calculations, we are still forced to view our task as accumulating evidence that the code is solving the particular problem correctly and accurately, or providing a demonstration of lack of evidence of the opposite conclusion.

The AIAA Guide [12,242] defines *error* to be "A recognizable deficiency in any phase or activity of modeling and simulations that is not due to lack of knowledge." This definition stresses the feature that the deficiency is identifiable or knowable upon examination; that is, the deficiency is not caused by lack of knowledge. This definition leads to the so-called *unacknowledged error*, one of two types of errors identified in the AIAA Guide. Unacknowledged errors are blunders or mistakes, such as programming errors, input data errors, and compiler errors. There are no straightforward methods for estimating, bounding, or ordering the contributions of unacknowledged errors.

The second type of error identified in the AIAA Guide is an *acknowledged error*, which is characterized by knowledge of divergence from an approach or ideal condition that is considered to be a baseline for accuracy. Examples of acknowledged errors are finite precision arithmetic in a computer, approximations made to simplify the modeling of a physical process, and conversion of PDEs into discrete equations. An acknowledged error (simply called error from now on) can therefore certainly be measured in principle because its origins are fully identified. Quantification of error can proceed by comparison with a test problem or series of test problems. It might also be the case that quantification of the error cannot be achieved for reasons that have nothing to do with our ability to recognize that the error is present. For example, assuming we know that a specific discretization has introduced error, unless we can perform a detailed grid convergence study, we have no quantitative basis for estimating that error. Convergence studies must be performed in this case.

If divergence from the correct or more accurate approach or condition is observed by one means or another, the divergence may be either corrected or allowed to remain in the model. It may be allowed to remain because of practical constraints, such as the error is acceptable given the requirements. Or, it may be impossible to correct the error because the cost may be too great. External factors always influence whether or not it is acceptable to allow a known error to remain for a specific calculation. Given this possibility, the fundamental goal of verification assessment is still the identification and quantification of acknowledged errors and identification of unacknowledged errors in the computational code and its solution.

It is worth emphasizing the importance of error quantification and error estimation for verification of calculations in order to reach an understanding of the proper role of error in verification assessment. We begin by identifying exactly what we mean by error in the context of CFD calculations. Error in a CFD calculation typically is defined as

$$E = u_{\text{exact}} - u_{\text{discrete}}. \tag{1}$$

Here, u_{exact} is the mathematically correct solution of the exact equations of the conceptual model PDEs for a given set of initial and boundary conditions. U_{discrete} is the numerical solution of a given discrete approximation of these same PDEs and set of initial and boundary conditions produced by the given code on a given computer. We assume for this discussion that u_{exact} and u_{discrete} are scalar fields. E depends on the discretization, symbolized by the characteristic mesh spacing h, which may be one-, two-, or three-dimensional and which may be spatially varying and time-dependent in adaptive discretization, the temporal discretization τ, and possibly iteration parameters which we do not emphasize specifically in the following notation. E always depends on the particular problem chosen for study by a CFD calculation.

It is useful to introduce a further conceptualization. Let $u_{h,\tau\to 0}$ be the limit as $h\to 0$ and $\tau\to 0$ of the exact solution of the discrete mapping of the PDEs and their initial and boundary conditions. Using the triangle inequality, estimation of E can be written as

$$E \leqslant \|u_{\text{exact}} - u_{h,\tau\to 0}\| + \|u_{h,\tau\to 0} - u_{h,\tau,I,c}\| < \varepsilon, \tag{2}$$

where $u_{h,\tau,I,c}$ is the discrete solution using a given algorithm for a given finite spatial grid h, time step τ, iterative convergence parameter I, and computer c. ε is an arbitrary small number, typically the choice of accuracy bound for the numerical solution. The norm symbols used in Eq. (2) are suggestive of differences between the functions only. These differences may well be measured by the application of appropriate function space norms. But this is not necessary—it could be as simple as an absolute value at a specific space-time point.

The first term in Eq. (2), $\|u_{\text{exact}} - u_{h,\tau\to 0}\|$, is the error introduced by the exact solution of the discrete equations. In principle, it can be argued that this term is zero from the standard principles that underlie the development of discrete algorithms for solving the PDEs. For example, it is commonly zero for strongly consistent (see Section 3.3.1), stable difference schemes. Trying to ensure that this term is zero is primarily the concern of the numerical analyst, algorithm developer, and mathematician.

For relatively simple flows and simple space and time discretization schemes, one can have confidence that the first term in Eq. (2) is zero. Generally, we can argue that this term is zero only for linear, constant coefficient PDEs and strongly consistent, stable numerical schemes. Few or none of these restrictions may ever apply to real application calculations such as for nonlinear, three-dimensional, time-dependent PDEs used to model strongly coupled multiscale physics and presenting chaotic or stochastic solutions. For example, Yee and Sweby [346] review a variety of simplified problems relevant to CFD that illustrate the difficulty of demonstrating that this term is truly zero.

The second term in Eq. (2), $\|u_{h,\tau\to 0} - u_{h,\tau,I,c}\|$, is quite another thing and is the primary reason for performing calculation verification. This term can only be estimated from empirical assessment, i.e., observed computational experiments. The second term will be nonzero due to finite h, τ, and I, finite precision arithmetic, and programming errors in the source code and operating system code (especially on the latest technology supercomputers). This term may not even be small in complex calculations because of contemporary computing resource limitations. The size of this term in Eq. (2) for resource-constrained values of h thus becomes very important. In point of fact, this term can never become exactly zero—at best it can only be zero to machine accuracy. (The influence of machine precision on computability of solutions to PDEs is beyond our scope. A good reference to this subject is by Chaitin-Chatelin and Frayssé [62].) A posteriori error estimates are likely to be most relevant to the estimation of the second term in Eq. (2).

Dealing with the second term in Eq. (2) is mainly an enterprise for verification of calculations (solution accuracy assessment), although code verification elements such as SQE (software quality engineering) aid in minimizing the possibility of unacknowledged error contributions. Calculation verification concentrates on bounding $\|u_{h,\tau\to 0} - u_{h,\tau,I,c}\|$ as precisely as possible for specific problems. It forces us to make a fixed choice of application and examine particular computational performance in order to bound the term. In an analytic test problem, for example, which is really the only case where we are likely to know both u_{exact} and $u_{h,\tau\to 0}$, we often do not estimate this term on the entire spatial and temporal domain, but only at a specific subset of that domain, such as a single space-time point.

The restrictions that are required to have a chance of dealing with the error in Eq. (2) have powerful implications for test-centric verification. Software engineering is important when establishing an estimate like that in Eq. (2) because the software should be properly constructed to justify this effort. Empirical evidence of software correctness is often assumed to accumulate by assessing the error for specific calculations, but this does not constitute real proof that the software is correct. For example, the fortuitous cancellation of two large errors could confirm the estimate like that in Eq. (2) for a

specific calculation, yet the software would still be wrong. How much weight such empirical evidence should be given in verification assessment is always a question.

3.3. Role of computational error estimation in verification testing

3.3.1. Convergence of discretizations

The discrete approximation to the PDEs, for example with finite-difference or finite-volume approximations, introduces *truncation error* [191]. *Discretization error*, on the other hand, is the error introduced by the solution of the discrete equations when contrasted with the exact solution of the PDEs. A discretization is defined to be *strongly consistent* [191] if the discretization converges to the PDEs with zero truncation error in the limit of zero grid spacing and time step. Since truncation error estimates typically involve derivatives evaluated at grid points, such convergence need not be true for equations with discontinuous solutions (shocks and contact discontinuities, for example). Laney [191] mentions a variety of discretizations of the Euler equations that do not have the property of strong consistency.

For strongly consistent schemes, in the limit of sufficiently fine grids and under simplifying assumptions (e.g., linearity, constant grid spacing), heuristic arguments can be presented that the discretization error is proportional to the truncation error (see, for example, [115]). The truncation error for strongly consistent discretizations is $O(h^p, \tau^q)$, $p > 0$, $q > 0$, where we assume one space dimension. This is a convenient starting point for estimating discretization error if the discrete solutions converge. Then, for sufficiently small h and τ, we have

$$\|u_{\text{exact}} - u_{h,\tau}\| = O(h^p, \tau^q). \tag{3}$$

Eq. (3) is understood to be a local estimate if spatial grid size varies over the domain of the discretization. The norm in Eq. (3) is any appropriate measure of the difference between solutions of the discretization and exact equations. For example, it could simply be the absolute value of a pointwise difference. Further discussion of the relationship of truncation error and discretization error with a code verification focus is presented in [46].

The role of discretization error in our considerations of verification is best described in terms of the following questions that must be answered.

- Does the discrete solution converge to the exact solution as the mesh spacing is reduced in real calculations?
- What is the effective order (the observed values of p and q) of the discretization that is actually observed in calculations?
- What is the discretization error that is actually observed for real calculations on finite grids?

We use the terms *a priori information and a posteriori information* in this paper and need to briefly explain what we mean by these terms. Usually, the term a priori in mathematics is associated with the development of estimates for PDEs that facilitate the proof of existence theorems (see [111] for a simple example of the application of energy estimates to prove the existence of solutions to linear evolution equations), as well as theorems concerning the smoothness of solutions (see [119] for an example). These estimates are called a priori because they use only information about the partial differential operators and the associated initial and boundary data. A priori estimates do not rely upon direct knowledge of the properties of the solution of the PDE.

Our use of the term a priori information loosens the traditional restriction of this term to include any information concerning the numerical algorithm that relies only upon the partial differential operator, its initial and boundary data. Classical a priori inequalities may be directly useful in deducing such information, such as the application of energy inequalities to prove the stability of certain difference schemes for nonconstant coefficient PDEs [271]. But we also include a classic result like the Lax Equivalence Theorem (see below) as an example of a priori information (although this theorem does not rely upon classical a priori inequalities for its proof).

A posteriori information, on the other hand, does rely upon knowledge of the solution of the relevant PDEs. For example, this information may take the detailed form of a posteriori error estimates [14] that can be used, say, to control adaptive grid strategies. Our perspective in this paper is somewhat more informal. By a posteriori we mean "empirical," unless otherwise specified. In other words, a posteriori information is information that is gathered by examination of the results of specific numerical calculations, or "observed" information as Roache [278] calls it. This information could be as specific as an estimate, or it may be direct observation of the convergence characteristics of a finite difference scheme on a series of refined grids. The Richardson h-extrapolation method discussed below is an example of using a posteriori information. The key point in our minds is that the information is gathered from numerical calculations, not only from the properties of the underlying partial differential operators and their associated discretizations.

To address these questions, the most important activity in verification testing of codes is the systematic refinement of the grid size and time step for specific problems. If the reasoning that relates truncation error to discretization error for strongly consistent schemes is

correct, including convergence of the discrete solution, then the discretization error should asymptotically approach zero as the grid size and time step approach zero, exclusive of computer round-off errors. If the order of accuracy is constant as the grid and time step are reduced beyond a specific threshold, we say that we are in the *asymptotic region* of the discretization. Empirical studies in the asymptotic region are particularly important for resolving the key questions above. For example, when numerical performance in the asymptotic region has been demonstrated, Richardson's extrapolation can be used to estimate the converged discrete solution [45,61,113,117,270,278,353]. We focus attention on this issue below.

Because this definition of empirical convergence typically demands a large amount of computer resources, it is usually applied on simplified or model problems, which must certainly be elements of any set of verification tests for the code in question. An alternative, but certainly not rigorous, meaning of empirical convergence that is also used in practice is to observe little change in important dependent variables during the course of grid and time-step refinement studies. We emphasize that empirical grid and time-step refinement studies often expose discretization errors and programming errors in boundary conditions, as well as in the underlying PDE discretization. This is a key reason for worrying about a priori understanding of estimates like that in Eq. (3). When the deviation of empirical performance from such an estimate is not understood, this condition is often a symptom of coding errors, either in algorithm formulations or in programming or both. Roache [278] presents vignettes that demonstrate this concern.

Careful empirical assessment of iterative performance in a code is also important. In most practical application simulations, the equations are nonlinear and the vast majority of methods of solving these equations require iteration (for example, implicit methods for computing steady states in CFD). Iterations are typically required in two situations: (1) globally (over the entire domain) for boundary value problems, and (2) within each time step for initial-boundary value problems. Thus, iterative convergence is as much of an empirical issue as space-time grid convergence. The questions raised above for space-time convergence have exact analogs for iteration schemes: Does the iteration scheme converge and does it converge to the correct solution?

Often a scaled iterative-convergence tolerance is specified, and the difference between the solution of successive iteration steps at each point in the grid is computed. If the magnitude of this difference is less than the specified tolerance, then the numerical scheme is defined to be iteratively converged. Scaling, of course, cannot be done when the scaling value is pathologically close to zero in finite arithmetic. Then other measures of iterative convergence must be introduced, as in the residual example below. In verification testing, the sensitivity of the solution to the magnitude of the convergence criteria should be varied and a value should be established that is justifiably consistent with the objectives of the simulation. It should be clearly realized that such convergence criteria, both absolute and relative errors, depend on the rate of convergence of the iterative scheme.

For pure (time-independent) boundary value problems, a more reliable technique of determining iterative convergence is to base the criteria on the residual error that remains in the approximate solution to the difference equation [21]. A residual vector is computed for each iteration, i. e., the error in the present solution iteration as compared to the exact solution of the difference equations. To measure the residual error over the entire domain, an appropriate vector norm is computed. This value is then compared with the magnitude of the residual error at the beginning of the iteration. When the error norm decreases by, say, five or six orders of magnitude (assuming that the initial error is $O(1)$) one can more confidently determine iterative convergence. This error norm can be computed for any or all of the equations in the system of PDEs. This technique of computing the residual error is applicable to a wide variety of iterative methods and its reliability is not dependent on the rate of convergence of the numerical scheme. Iterative convergence errors for steady state problems are discussed in Ferziger and Peric [115] and in Roy and Blottner [285].

For unsteady problems like hyperbolic problems, an implicit solution is usually obtained by marching in time and globally solving in space at each time step. The iterative procedure is similar at each time step; however, since time-step error is cumulative, iteration errors can accumulate and destroy the integrity of the solution. For an unsteady problem, the values of relative per-step convergence criteria should be at least an order of magnitude smaller than the global convergence criteria for a steady-state problem. Typically, in practical unsteady problems, iterations cannot be performed to machine precision. The contribution of finite iteration-convergence errors to local and global errors in unsteady problems obviously then becomes complicated. For purposes of verification, each distinct calculation must be assessed separately to study these contributions.

There are other issues associated with computational error that go beyond error estimates. We can ask fundamental questions of how well the computational world matches the "true" dynamics of the underlying equations. For example, how well are the dynamical system features of the underlying differential equations matched by the computational dynamics? In the case of steady flows, this question is concerned with the stability and bifurcation behavior of the computed solutions.

Attempting to measure the computational "error" in this case is very difficult. For example, it is only in relatively simple cases that the "true" stability and bifurcation behavior may be well understood for the conceptual model. An entire taxonomy of potential threats to computational accuracy arises in this point of view. Potential dangers are summarized in [346] and the work cited there.

3.3.2. A priori error information

We emphasize that by "a priori," we mean information, such as an estimate like that in Eq. (3), that is developed without information resulting from direct observation of the code's numerical performance. As we noted previously, this topic is a significant element of classical numerical analysis for PDEs, especially those underlying CFD [115,158,159,191,199,271], and is important information for code verification [278]. Here, our intent is to emphasize the assumptions that enter into a priori information development and the influence these assumptions will have on verification assessment. Our basic point in this discussion is that a priori error estimates based in numerical analysis serve primarily as guides in verification activities. The only quantitative assessment of numerical error that we can achieve in almost all cases is through a posteriori error estimates.

Strong consistency is the most fundamental a priori information that is required for verification. Discretizations that are not strongly consistent will not converge, and the three major questions for empirical performance assessment will not make sense. Consistency is required for the first term in Eq. (2) to have the possibility of being zero. As emphasized above, even consistency has underlying assumptions to achieve a priori estimates of discretization convergence, such as solution smoothness for the PDEs. Dealing with singularities of various types in solutions of PDEs introduces difficulties, especially in higher dimensions, that cast in doubt the likely empirical accuracy, or even applicability, of a truncation-error–based estimate of discretization error like that in Eq. (3).

The assumption of smoothness of the solution in establishing consistency of discretizations is serious. Especially for a wide variety of general transient compressible flows, the assumption of smoothness is not correct—a fact that often has the empirical consequence of reducing the order of the discretization error below the a priori estimate of the discretization given in Eq. (3), even in regions well removed from the singularity. The "pollution" of particular regions of a calculation by the presence of singularities such as shock waves or geometrical singularities is a subject of grave concern in verification, even for application of an error estimate like that in Eq. (3) in a local sense for grid adaptivity. Often, the only clear sense of this pollution available to us is through careful empirical assessment. Since the technical aspects of this issue are beyond the scope of this paper, the reader should consult a series of papers by Babuska and colleagues [24,25,26] as well as the paper by Oden [247] for a discussion of this problem from a finite-element point of view. Recent work of Zhang and his colleagues [350] discusses the effect of the presence of a shock wave structure on a posteriori error estimates for the Euler equations; see [278] for additional references on CFD code verification with shocks. A recent paper of Botella and Peyret [48] discusses similar problems associated with computing singular solutions of the Navier–Stokes equations.

We have also had to make the implicit assumption that the spatial mesh is uniform to arrive at the result given in Eq. (3) if it is intended as a global error estimate. This assumption is virtually never correct in real calculations. In addition, for unsteady flow calculations, the temporal spacing is not uniform either. Formal estimates of global truncation errors are difficult to derive on inhomogeneous meshes. At the very least, this means that Eq. (3) can at best be interpreted as a local error estimate. As pointed out by Laney [191], the effective global order of accuracy that results on variable grids empirically tends to be less than such a local estimate. Heuristically, it is also known that direct application of a truncation error estimate for a discretization error estimate such as that in Eq. (3), called the *formal order* of the discretization error, tends to be greater than a "true" local discretization error. The point is that sorting out what errors are really being predicted by a priori estimates influences the kind of verification information we achieve by using those estimates.

We conclude by making a point that is relevant to understanding convergence and discretization error for verification. As we have stated above, application of formal truncation error estimates to a priori estimates of discretization error for strongly consistent discretizations implicitly requires solution convergence of the discretization. Consistency is a necessary condition for solution convergence, but it is not a sufficient condition. Consistency certainly does not imply convergence of the discretization to the "correct" solution of the PDEs, even assuming that there is a unique solution. The supplementary condition that is required for solution convergence is *stability* of the discretization.

Only one general theorem confirms that stable, consistent schemes are indeed solution convergent. This is the famous result due to Lax (see [271] for a rigorous proof of this result).

Lax Equivalence Theorem: Given a properly posed linear initial-value problem for a hyperbolic PDE, and a finite-difference approximation to it that satisfies consistency, then the approximation converges to the correct solution if and only if the approximation is stable.

The assumption of linearity required to prove the Lax Equivalence Theorem is a strong one. There is no analog of the Lax Equivalence Theorem for nonlinear problems. However, linearity restrictions might still imply useful information as long as the resulting error estimate like that in Eq. (3) is intended to be local, not global. The appropriate heuristic is then that necessary and sufficient conditions for solution convergence of the discretization are consistency and stability even for nonlinear problems. There is plenty of empirical evidence that supports this heuristic. There are also some isolated rigorous results for specific PDEs. For example, see [199] for some theorems specific to conservation laws.

Demonstrations of stability for problems where the assumption of linearity is violated are important to more conclusively assert our heuristic above for nonlinear problems. The most dominant technique of stability analysis is von Neumann stability analysis, which is not only limited to linear problems but also limited to the assumption of a constant grid spacing. This last restriction can be overcome in specific circumstances, for example, through the use of specialized a priori estimates for PDEs like the energy method [271].

Laney [191] reviews various elements of nonlinear stability analysis primarily centered on concepts of monotonicity and total variation boundedness for conservation-law discretizations. The greatest restriction of this stability work is to one space dimension. There are great difficulties in extrapolating theoretical or numerical behavior from one spatial dimension to two or three space dimensions for the basic equations of CFD (Euler and Navier–Stokes equations). Once again, this community has traditionally relied upon empirical evidence of performance to support such extensions, with consequent reduction in the quasi-rigorous underpinning of understanding of solution convergence of the discretizations. This worry is primarily a concern for numerical analysis, not verification, at this point in time. But it is relevant to an understanding of the limitations involved with, for example, extrapolating verification results in one-dimensional calculations to higher dimensions.

3.3.3. A posteriori error estimates

Spatial and temporal convergence is fundamentally an asymptotic concept. From this perspective, the only real confidence we can have in a calculation is if we can achieve sufficient discretization resolution, both temporal and spatial, to directly assess convergence. In other words, from an asymptotic-convergence point of view, the only definitive statement about computational error that we can make is that

$$\|u^{r1} - u^{r2}\| \leqslant \varepsilon, \tag{4}$$

where $r1$ is one level of temporal and spatial refinement of the discretization, $r2$ is a more refined level, and ε is the accuracy required of the calculation. Eq. (4) assumes that the convergence implied by Eq. (3), when the discretization scheme is stable, is eventually monotonic and that we have entered the asymptotic regime. The ability to converge given computations in the sense of Eq. (3) would solve all of our practical problems. Unfortunately, the full convergence of a numerical computation implied by Eq. (3) can hardly ever be attained in complex realistic simulations, simply because the computational resources required to achieve convergence in typical difficult applications problems are still not available.

We discuss a posteriori error estimation through the use of Richardson extrapolation [278]. An elegant and concise discussion of this topic within a larger context of characterizing modeling uncertainties can also be found in [17]. To illustrate the challenges, we focus on grid convergence extrapolation as discussed in [17]. Consider for simplicity a steady-state computational problem in one spatial dimension with uniform mesh spacing h. We must make several assumptions in order to perform Richardson extrapolation in a straightforward way:

- Assumption 1: u is a smooth solution (existence of sufficient derivatives to justify the application of a Taylor expansion in the mesh spacing) of the exact PDEs.
- Assumption 2: The formal convergence order p of the spatial discretization method is known a priori. In addition, it is assumed that the computer code has empirically demonstrated convergence order p.
- Assumption 3: The mesh spacing is small enough that the leading-order error term dominates the total discretization error. This is also called the *asymptotic range* of the discretization [278]. One key implication of this is that convergence is monotone in the asymptotic range.

Thus, under these assumptions, expand the exact solution of the PDE of interest as

$$u_{\text{exact}}(\vec{x}) = u_h(\vec{x}) + \alpha h^p + O(h^{p+1}). \tag{5}$$

α is a constant, while all other symbols are as defined previously. If u_h and p are known, then two numerical solutions with different grid sizes (different choices of h) are required to compute the two unknowns u_{exact} and α in Eq. (5). From this determination the discretization error can then be estimated from the two numerical solutions. If p is not known a priori, then it can be estimated using a similar approach, thus relaxing Assumption 2. We discuss this below.

Following [17], apply Eq. (5) to two different nested grids (assume that $(h_2/h_1) < 1$):

$$\begin{aligned} u_{\text{exact}} &= u_{h_1} + \alpha h_1^p + O(h_1^{p+1}), \\ u_{\text{exact}} &= u_{h_2} + \alpha h_2^p + O(h_2^{p+1}). \end{aligned} \tag{6}$$

We then find that

$$\alpha = \left[\frac{u_{h_1} + u_{h_2}}{h_2^p - h_1^p}\right] + O(h_1^{p+1}) + O(h_2^{p+1}), \tag{7}$$

$$u_{\text{exact}} = \left[\frac{h_2^p u_{h_1} - h_1^p u_{h_2}}{h_2^p - h_1^p}\right] + O(h_1^{p+1}) + O(h_2^{p+1}). \tag{8}$$

The term in brackets in Eq. (8) represents an extrapolation of the discrete solution toward the exact solution that is formally one order of accuracy higher than p. For example, in the special case of centered difference schemes, $p = 2$, it turns out the this extrapolation is fourth order. This is the original h-extrapolation method of Richardson [270]. (See also [45,61,240,278,309,338].)

Discretization error for a given solution, say u_{h_1}, can then be estimated as

$$\| u_{\text{exact}} - u_{h_1} \| \cong \left| \left[\frac{u_{h_2} - u_{h_1}}{h_2^p - h_1^p}\right] h_1^p \right| + O(h_1^{p+1}) + O(h_2^{p+1}). \tag{9}$$

Eq. (9) is an a posteriori error estimate. This is true for any value of h_1 and h_2 in the asymptotic range. Formally, the development of this error estimate can be extended to multiple spatial dimensions and time-dependent problems.

The common misuse of Richardson's method is the analyst's failure to *demonstrate* the validity of each of the assumptions stated earlier on the problem of interest and on the solution variable of interest. It must be recognized that a numerical method of formal accuracy p, does not necessarily mean that the computational result will also be of order p. We give two examples of how computational order of accuracy can be less than formal accuracy. First, a common cause for misuse of Richardson's method is failure of the analyst to demonstrate that the grid is fine enough to define the particular solution variable of interest to be in the asymptotic convergence region. The *only* way asymptotic convergence can be demonstrated is to compute *three* solutions, each with a different spatial resolution and each with the same refinement ratio. The grids do not have to be halved in each spatial direction, but it is common to do so. The procedure is as follows: The fine and medium grid solutions are assumed to be in the asymptotic region, and the equation above is used to compute the exact solution. Then the error on the coarse grid is computed by using the coarse grid solution and the just-computed exact solution. If the error on the coarse grid solution does not closely match the expected asymptotic error for the given grid refinements, then the solution on the coarse grid is demonstrated *not* to be in the asymptotic region. For example, if the grid is halved in each spatial direction for each refinement (Δ, $\Delta/2$, and $\Delta/4$), then for a second-order accurate method the coarse grid solution should have an error of 16 times the error of the fine grid solution.

A second cause that can reduce the effective order of accuracy of a code is that a programming error in the code, either for field points or boundary points, can degrade the formal accuracy of the method. Also, if the numerical implementation of Neumann boundary conditions does not have the same formal accuracy as the field-point equations, then the computational order of accuracy will fall in between each of the formal accuracies. The computational accuracy will vary over the field, depending on the proximity of the points to the boundaries. That is, far from the boundaries the computational result would be near the field-point accuracy, while near the boundaries the accuracy would be near the numerical accuracy of the boundary conditions. This same characteristic will occur for numerical methods that have variable order as a result of local flow phenomena, such as first-order shock-capturing methods and variable-order upwind methods. The impact of boundary conditions on the effective order of a numerical scheme is an area that requires more research. Blottner [44] shows that for inviscid supersonic flow the effects of outflow boundary condition errors are essentially local; for diffusion problems, on the other hand, Blotttner shows boundary condition errors have global influence. Srivastava, Werle, and Davis [319] also demonstrate that boundary *curvature* errors can be very important.

Singularities and discontinuities significantly complicate the process of verification and certainly influence extrapolation. By definition, the discretization is not valid near a singularity because higher-order derivatives that are neglected in the Taylor series expansion are not small. A suitable mathematical transformation should be used, when possible, to remove singularities that are caused by the geometry or the coordinate system. Singularities that are inherent in the conceptual model should be removed, if possible, by including the appropriate physical information that was left out of the discretized model. In problems where the singularity cannot be removed and in flow fields with discontinuities, it is to be expected that local grid and time-step refinement may not lead to a fully grid-resolved solution. For such problems, one should present the results of the local grid and time-step refinement and should document the extent to which the singularity and discontinuity influenced the grid and time-step refinement elsewhere in the flow. Recently, Carpenter and Casper [57] carefully examined the issue of order of convergence of shock-capturing schemes and arrived at the following conclusion: "This study shows, contrary to conventional wisdom, that captured two-dimensional shocks are asymptotically first order, regardless of the design accuracy of the numerical method." This is a sobering result, not only with regard to determining the order of accuracy in verification analyses, but also in practical applications of CFD.

Recently, another relaxation of the fundamental assumptions underlying Richardson extrapolation has been reported. In Roy et al. [286], the assumption of the grid being in the asymptotic region of the discretization is relaxed for verifying calculations of hypersonic flow over spherically blunted cones. This is part of a larger effort to validate a code capability for computing such flows. In this work, the authors assume that both first- and second-order errors are present in the discretization error for the specific grid resolutions of interest and develop an estimate for the exact solution using extrapolation. There is a shock wave in this flow field, so that the assumption about smooth solutions is also violated. Awareness of the importance of higher order terms in understanding the behavior of Richardson extrapolation results has been known for some time (Blottner [43]).

Three discrete solutions, using the same grid nesting as in Eq. (8) above, are required to estimate the exact solution in the hypersonic flow case. The result reported in the paper of Roy et al. is

$$u_{\text{exact}} = u_{h_1} - g_1 h_1 - g_2 h_1^2,$$
$$g_1 = \frac{u_{h_3} - u_{h_2} - r^2(r^2-1)u_{h_2}h_1^2}{r(r-1)h_1},$$
$$g_2 = \frac{u_{h_3} - u_{h_2} - r(u_{h_2} - u_{h_1})}{r(r-1)(r^2-1)h_1^2}. \tag{10}$$

Basically, the import of this work is that in mixed-order discretizations, the higher-order accurate terms dominate on relatively coarse grids, while the lower-order terms (e.g., first order in shock capturing) dominate on fine grids. Fig. 5 illustrates a key result from this paper.

We can generally relax the underlying assumption of knowing the correct order of the discretization error. Instead, we can estimate the observed order of the discretization error by using Richardson extrapolation. For purposes of verification, in fact, it is probably advisable to always ascertain the empirical error order and use it for grid extrapolation. As noted in [278] (and originally due to [89]), by using calculations on three separate grids and assuming that we are in the asymptotic regime, the empirical order $\tilde{p}$ of the discretization is

$$\tilde{p} = \ln\left(\frac{u_{h_3} - u_{h_2}}{u_{h_2} - u_{h_1}}\right) / \ln(r),$$
$$r = (h_3/h_2) = (h_2/h_1). \tag{11}$$

How practical such an estimate is, in the sense of whether one can afford to perform the convergence studies, always depends on the specific calculation one is examining.

In our discussion here we have concentrated on the use of Richardson's method to estimate grid and time-step convergence error. Richardson's method can be applied to both finite difference methods and finite element methods. Although a Richardson-like extrapolation error estimate like that in Eq. (9) can also be developed for finite element calculations, specific a posteriori error estimators are also available in the finite element literature. An explicit discussion of these estimators is beyond the scope of this paper; see [17] for a brief description.

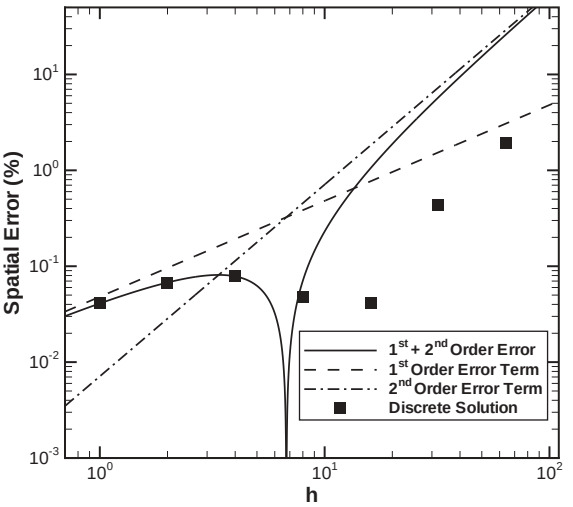

Fig. 5. Demonstration of extrapolated error estimation for mixed first and second order schemes [286].

A posteriori error estimates are important for finite element adaptivity, where both the spatial grid density (*h-adaptivity*) and the order of the finite element scheme (*p-adaptivity*) can be adapted. The important role of a posteriori estimation in adaptivity for finite elements is discussed in many papers, as well as the recent book by Ainsworth and Oden [14]. For verification and validation, however, the real issue is accurate error estimation, not element adaptivity. As pointed out by [278], the use of Richardson's method produces different estimates of error and uses different norms than the traditional a posteriori error methods used in finite elements [159,274]. Although traditional a posteriori methods used in finite elements are very useful and computationally efficient, they do not demonstrate asymptotic convergence or directly address useful engineering error measures. Recent work in a posteriori estimation addresses error in local quantities of engineering interest for certain types of nonlinearities in elliptic problems (see, for example, Ref. [14]).

Richardson's extrapolation can be applied to compute dependent variables at all grid points as well as solution functionals. Solution functionals are integrated and differentiated quantities such as body lift and surface heat flux, respectively. Venditti and Darmofal [332] discuss the coupling of error estimation and adaptive

gridding to reduce numerical errors in such computed functionals. Their particular example is the quasi-one-dimensional flow in a variable-area duct. It is important to emphasize that different dependent variables and functionals converge at different rates. For example, the grid and time step that are required to show second-order convergence in heat flux on a body surface are typically much finer than for total lift on a body. A Grid Convergence Index (GCI), based on Richardson's extrapolation, has been developed to assist in the estimation of grid convergence error [275,277,278]. The GCI converts error estimates that are obtained from any grid refinement ratio into an equivalent grid-doubling estimate.

Finally, in our opinion there may still be practical issues in performing calculations on suitably nested grids for developing an estimate like that in Eq. (9). One simple illustration is the following thought experiment. Consider a large calculation, one that is at the limit of available computational resources. It will be infeasible to refine the grid for such a problem. Roache has pointed out that to use Eq. (9) one does not necessarily need to refine the grid. Rather, if one is in the situation described, one could coarsen the grid instead to achieve the needed two refinement levels. However, suppose that the large calculation resulted from the requirement to resolve an important flow structure (shock, reacting flow region, turbulent mixing layer) with the *minimal* needed resolution. Or, alternatively, suppose that a lower resolution grid does not lie in the asymptotic regime of the discretization. This is the case in certain compressible flow problems, where all of the overall grid resolution is driven by the need to resolve a captured shock with four, say, grid points. In such a case, one cannot coarsen the grid because one encounters a qualitative change in the computed flow upon doing this, or one violates the key assumptions underlying the extrapolation. One is essentially constrained to work with one grid. The development of a posteriori error estimators for single grids is thus of interest.

There is some question whether or not assessing numerical performance on grids that are known to be unconverged, or underresolved, is properly in the domain of verification assessment, specifically calculation verification. Our point of view on this matter is completely practical. Typically, calculations for complex applications are performed on grids that are known to be underresolved, at least in some regions. While one may in principle apply extrapolation techniques to understand the converged solution, in practice this is problematic for many kinds of calculations because of the ease with which the underlying assumptions necessary for extrapolation are violated. This forces us to assess the quality of calculations on such grids.

Extracting an estimate of numerical accuracy on underresolved grids is a topic of current research. We previously mentioned one aspect of the problem, generically referred to as *pollution error*—the propagation of discretization error from less accurately resolved regions of a calculation into more accurately resolved regions. Another issue is simply to understand what information can rationally be synthesized from calculations with more or less known resolution problems. Though this topic is beyond the scope of this article, some references for the interested reader that demonstrate the need for research on this topic are [68,69,70,132,133]. These papers demonstrate that techniques for understanding the numerical accuracy of underresolved calculations are challenging and specialized. A critical theme in this literature, and one that might be generalizable to other important problems, is the role of statistical estimation in performing such assessment. We conclude by commenting that studying verification and validation for underresolved calculations directly addresses challenges posed by Gustafson in his 1998 paper [143].

3.4. Testing

3.4.1. Need for verification testing

Verification testing is a critical component of verification assessment in CFD. To the disbelief of many, a recent comprehensive analysis of the quality of scientific software by Hatton documented a dismal picture [149]. Hatton studied more than 100 scientific codes over a period of 7 years using both static and dynamic testing. The codes were submitted primarily by companies, but also by government agencies and universities from around the world. These codes covered 40 application areas, including graphics, nuclear engineering, mechanical engineering, chemical engineering, civil engineering, communications, databases, medical systems, and aerospace. Both safety-critical and nonsafety-critical codes were comprehensively represented. All codes were "mature" in the sense that the codes were regularly used by their intended users, i.e., they were codes that were approved for production use. The total number of lines of code analyzed in Fortran 66 and 77 was 1.7 million, and the total number of lines analyzed in C was 1.4 million. For these codes, Hatton conducted what he calls the T experiments: T1 tests were static analysis tests relying on complexity analysis and safe programming standards; T2 tests were dynamic analysis tests that were, interestingly enough, glass-box tests to a certain extent. A definition and discussion of glass-box testing is presented in Section 3.4.2.

Some of the major conclusions from this comprehensive study were as follows:

- There were about 8 serious static faults per 1000 lines of executable lines in C, and about 12 serious faults per 1000 lines in Fortran. A fault (Hatton's terminology) is not something that necessarily prevents the

code from executing. Rather, a fault causes an incorrect answer, even when the code executes without an obvious crash of some kind. It is generally understood that successful software execution that likely produces moderately incorrect results is the most damaging possible problem in software because it is almost impossible to discover and thus fix.

- The T2 experiments provided very disturbing information in one application area—an area that emerged better than average in terms of the T1 experiments. In the T2 experiments, observed disagreement between nine different implementations of the same published mathematical algorithms, which were written in the same language and used the same input data, revealed that the computed output values of the codes agreed to only one or two significant figures. This study was particularly interesting because glass-box knowledge was used to generate the tests that provided this information.
- The Fortran-written codes were on average about 2.5 times longer than their C counterparts for solving the same type of problem, and the Fortran-coded functions had correspondingly more parameters that were passed as arguments than did their C counterparts. Hatton went out of his way, however, to stress his belief that no programming language used in scientific code development was immune from the problems uncovered in his work.
- Hatton's major conclusion was:

  "The T experiments suggest that the results of scientific calculations carried out by many software packages should be treated with the same measure of disbelief researchers have traditionally attached to the results of unconfirmed physical experiments."

We particularly like the way this last conclusion is stated because it emphasizes one of our most important conclusions in this paper—that properly executed verification (and validation) in CFD should be held to the same standards used for verification and validation of physical scientific experimentation. Hatton's conclusion is disappointing, but not surprising in our view. The somewhat polemical style of Hatton's paper has somewhat diluted the influence of his conclusions.

As a side note, we also mention that this study is one of the best that we have seen in the published literature on how to do reasonably objective code comparisons. We also observe that Stevenson [321] strongly agrees with Hatton's view that the problems uncovered by his static testing experiments are basically independent of programming language, as does Gustafson [143]. The problems are structural to the approaches used to develop CFD codes, not to the specific programming implementation details. Reading between the lines, it appears that the most recent study of these matters by Ambrosiano and Peterson [18] is also essentially supportive of this position.

We believe there are several reasons why formal testing of CFD software is not a common practice. The first reason is that formal testing processes are traditionally associated with software implementation assessment, whereas the emphasis in traditional CFD testing is numerical performance for correct implementations. The second reason revisits one raised previously, namely that people working on CFD codes do not like to view themselves, or their work, as software engineers. A third reason is that there is less perceived risk associated with incorrect function of most or all CFD codes than there is for other kinds of software, such as safety-critical systems. For example, Johnson [172] discusses rigorous approaches to developing fault-minimal and fault-tolerant avionics software. There are regulatory requirements governing software verification for avionics applications [112] that are driven by the safety-critical consequences of software failure. We have never encountered a similar paper in CFD that addresses the underlying consequences of software failure as they are addressed by Johnson.

3.4.2. Algorithm and software quality testing

Our view of the integration of a set of verification activities, including SQE activities as well as testing, into a CFD verification process is conceptually summarized in Fig. 6. In this figure we have depicted a top-down verification process for CFD. The process has two main branches, one defined as "code verification" and the other as "calculation verification." The code verification branch is primarily centered on SE (software engineering), which includes practices and standards associated with SQE (software quality engineering). A major component of these activities, in turn, is a focus on *software quality testing*. Software quality testing emphasizes programming correctness. Software quality testing, as shown in Fig. 6, naturally divides into static and dynamic testing. Dynamic testing further divides into such elements of common practice as regression testing, black-box testing, and glass-box testing.

The other major branch of CFD verification activities emphasized in Fig. 6 is *algorithm testing*. Algorithm testing focuses on numerical correctness and performance of algorithms, and has the underlying goal of calculation verification. The major components of this activity include the definition of appropriate test problems for assessing solution accuracy, as well as assessment procedures for judging numerical performance versus these tests. An appropriate approach to CFD verification activities, especially for high-consequence computing, should include all of the indicated elements on both sides of the diagram.

As stressed in Section 3.2, no matter how many tests and empirical experiments we perform, our ability to observe code performance is very limited. We have also argued that we will never prove that the software

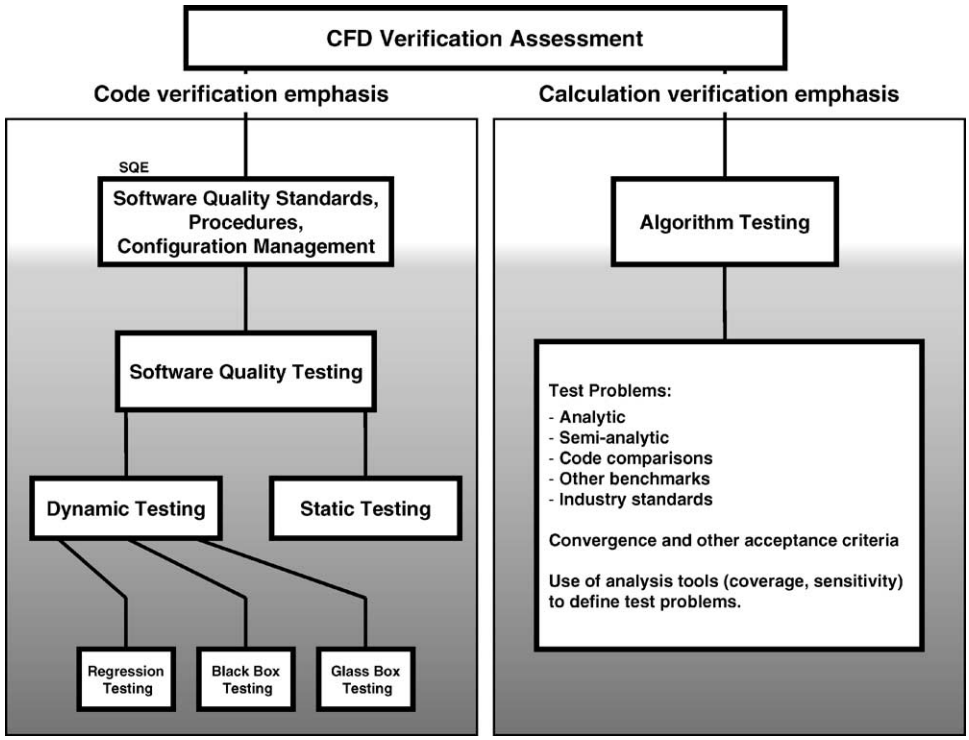

Fig. 6. Integrated view of verification assessment for CFD.

implementation is error free, or has zero software defects. In fact, as Hatton has conclusively demonstrated, one is always finding software bugs. Experience suggests that one can never totally eliminate bugs from complex CFD software, certainly not simply by performing algorithm testing alone. It seems to us that the best approach for minimizing the number of bugs that may be present in a CFD code at any given time is to seriously attend to software quality testing, along with coordinated SQE processes that are associated with the CFD code development activities, in partnership with detailed algorithm testing. Importantly, software quality testing does not eliminate the fundamental need for algorithm testing in CFD codes.

Software quality testing rests in three techniques: static analysis, dynamic analysis, and formal analysis [339]. Static analysis (or testing) techniques analyze the form, structure, and consistency of the code without executing the code. Software reviews, inspections, audits, and data flow analyses are examples of static analysis techniques. Dynamic analysis (or testing) techniques involve execution of the code. The results of the code execution are analyzed to detect coding errors or weaknesses in design that can cause coding errors. Regression testing, which re-evaluates the accuracy of computed results, is in the dynamic analysis category. Further remarks on static, dynamic, and regression testing are included below. We touched on the use of formal methods earlier and will not discuss them further.

It is probably safe to claim that static testing has traditionally been dominated by the compilation of code and the detection of compilation errors. One doesn't tend to see published studies of, say, the application of architectural analysis, such as the complexity metrics applied in Hatton's analysis [149], to CFD codes. We strongly advocate the expansion of static analysis as part of verification assessment for CFD codes. The complexity of modern programming languages (C++ especially) and the use of massively parallel computing platforms for CFD increase the need for and the impact of static analysis.

Systematic SQE testing literature, with its myriad paradigms for static and dynamic testing, tends to be somewhat disjoint from the published practice in CFD, where the emphasis is on dynamic algorithm testing. While we do not give a detailed review of SQE testing in this paper, in Section 3.4.4 we review a few essential issues that are particularly germane for coordinating SQE testing with algorithm testing. Two particularly useful references for readers who wish to learn in detail about the vast subject of SQE testing are Beizer [36] and Kaner et al. [175]. The viewpoint of these books is sufficiently different to add an interesting nuance to a subject as seemingly dry (to CFD practitioners) as SQE

testing. Beizer's book is somewhat more formal in its approach, while Kaner et al. are less structured in their approaches and recommendations. We highly recommend both books.

When considering software testing, it is important to understand the distinction between *black-box testing* and *glass-box testing*. Glass-box testing is practiced primarily by code developers because it assumes, by definition, sufficient knowledge of the design and architecture of the code to design, populate, and assess test plans based on this knowledge. It is sometimes called path testing, although this is properly one subset of this type of testing. Glass-box testing is also traditionally heavily weighted to what we have called SQE testing, although that is not mandatory. When we discuss static testing, test problems generated by coverage analysis and regression testing below, we will be discussing examples of glass-box testing.

Black-box testing (also called functional testing) is effectively the paradigm for algorithm testing as we defined it above. Black-box testing can be performed by anyone involved with a given CFD code, but it tends to be associated with independent testing entities. Certainly in the CFD world, much black-box testing is performed by users of the code. (If a developer of the code is also a user, for purposes of this discussion we emphasize their role as a *user*). Black-box testing requires no detailed knowledge of the code software, although such knowledge can certainly help. Even when the person(s) who designed the algorithms and implemented the code execute a test problem to investigate the empirical performance of the code, they are performing black-box testing. When a CFD algorithm is described in a publication, for example, and tested on one of the benchmarks we mention below, a black-box test has been performed. The goal of the authors in this case is to assess functionality and accuracy of output results, not to test specific software elements. When users test codes by running their favorite test problems prior to application, they are performing black-box testing. Users are an enormous resource for black-box testing. Black-box testing dominantly underlies our discussion of error assessment in Section 3.2. When we discuss dynamic testing, manufactured solutions, aspects of regression testing, statistical testing, and benchmarks below, we will be discussing issues particularly relevant to black-box testing.

The goal of a coordinated, optimal test strategy for a CFD code is fundamentally to suitably blend glass-box testing and black-box testing. Beizer [36] and Kaner et al. [175] address such coordination in great detail in their books.

One issue that is discussed further in Section 3.4.3 is the proper definition of effective test problems. *Code coverage analysis* and *code sensitivity analysis* are methods that can be used to assess the complexity of the intended application of the CFD code in order to design collections of test problems. Coverage analysis enables one to determine what components of the code enter into the real application calculation. The relative importance of the selected component contributions to that calculation is then determined via a sensitivity analysis. Ideally, one would start coverage analysis with a tool of sufficient capability to assess code lines and units exercised by the code in performing the desired calculation. For example, PureCoverage™ [266] is a commercial tool that detects lines of code executed during the operation of C++ software and reports this information in a useful form. A major goal of coverage analysis for algorithm testing is to understand execution paths as well as lines touched in the code in order to execute the intended application calculation. One then designs tests that specifically target these modules and paths. This is clearly a glass-box-testing strategy and is discussed extensively in both Beizer [36] and Kaner et al. [175]. While a discussion of sensitivity analysis is beyond the scope of this paper, the topic is brought up in the context of software quality testing in [232].

A natural partner of the coverage analysis methodology for software quality testing is *regression testing*. A widely used commercial testing technique, regression testing is defined by Beizer [36] as "any repetition of tests (usually after software or data change) intended to show that the software's behavior is unchanged except insofar as required by the change to the software or data." What this means in CFD practice is that a compendium of tests is assembled, a baseline for code performance against these tests is established, and the tests are run periodically. When the current performance is determined to have deviated from the baseline, either the decision is made that a bug has been introduced, or the regression test suite is base-lined again. There are two requirements that make this approach feasible. First and foremost, the test suite must run sufficiently fast so that the testing can be performed frequently. Second, the regression testing should be performed for every software modification. For multiple developers of a CFD code, for example, this means that no modified software is accepted before regression testing is performed.

Regression tests that are glass-box tests are often correlated with coverage analysis, and are designed to provide testing of as many lines of code as is feasible. Systematic, cyclic testing, say once a suitable set of test problems has been defined, is known to be important in code development and maintenance. Unfortunately, this requires a trade-off between available resources which influences the fidelity of the testing. For example, a high-fidelity test suite that takes longer than 5–10 h to execute is probably not practical for execution every day. Thus, such a suite cannot be used to demonstrate the stability of daily code builds.

We believe that regression testing should be considered as part of SQE testing. What regression testing measures is code stability, not code correctness. For example, one may take an algorithm verification problem and define its equivalent regression test to be a simplification that runs only a fraction of the time defined for the algorithm verification problem. That simplification could be the identical problem executed on a very coarse grid, or it could be the identical problem run for only a few code time-step cycles or iteration loops. The appropriate metric for success in regression testing is change from past baselining. Correct execution of the problem enters in only at the establishment of the baseline. In this sense, regression testing should not be considered to be algorithm testing.

The main use of regression testing is to minimize the effort devoted to fixing incorrect new software that is introduced into existing code in a software development project. There is a balance that must be determined between the effort devoted to regression testing and the effort devoted to finding and fixing incorrect modifications of the existing code at a later date. At the same time, regression testing aims to cover as much of the code as possible. Development and implementation of regression testing is thus closely related to issues of software coverage. In the CFD projects that we are familiar with, the anticipated coverage target for regression testing is typically on the order of 80%. This is partly because there are specific code elements that developers are willing to define as not requiring coverage. More importantly, these coverage thresholds are established in the attempt to prevent the effort devoted to regression testing from becoming larger than the effort required to fix introduced errors.

Regression testing requires considerable resources for performing the testing, evaluating the results, and maintaining the testing capability. Depending on the size of the regression test suite, significant time can still be spent for a code to successfully execute all of the problems. In addition, a large amount of human effort is required to maintain regression test suites. Since CFD codes are rarely frozen in their capability, every time that capability changes it will likely require re-establishing regression baselines. For codes that are changing frequently, the level of effort is even greater. Is regression testing worth this kind of effort? A important measure of the human effort involved in regression testing, as we suggested above, is the amount of time spent correcting incorrect new code versus the amount of time spent maintaining regression test suites. Information about how much time is required can only be determined by collecting data on the time and costs involved in both types of efforts. Unfortunately, currently we only have anecdotal data from the code projects with which we interact. That data suggests that the resources invested in regression testing are worth the investment. However, it should be noted that the data from those code projects that measured the amount of work devoted to fixing incorrect code modifications prior to the implementation of regression testing were not quantitatively gathered prior to the onset of regression testing. Once regression testing started, of course, we lost the opportunity to collect that data.

3.4.3. Algorithm testing

A plan for algorithm testing for a CFD code should clearly express three major types of content [262]. First, the goals and logic of the testing should be clearly expressed. Second, the methods and principles used to design tests for the plan, as well as their detailed description, should be described. Third, the nature of the comparison between the code and the required baseline test result should be precisely defined, along with the associated acceptance criteria. It is important to have precise definitions of what constitutes success or failure in executing the test problems.

Understanding the logic that is operating in defining and executing a test suite for a CFD code is the first pressing issue for design and execution of algorithm testing. The AIAA Guide [12], for example, suggests the following logical organization of testing for CFD codes: (1) tests with exact analytical solutions, (2) tests with semi-analytic (reduction to quadrature to simple ODEs [ordinary differential equations], etc.) solutions, and (3) benchmark solutions to the PDEs that may be representative of application complexity but that are suitable for code comparison exercises or other quantitative means of assessing code accuracy. The important goal achieved in this approach is the precise and quantitative definition of acceptance criteria for comparisons with such problems.

The second pressing issue that enters into the design and execution of algorithm testing is choice and implementation of test problems. There are several possible approaches for choosing algorithm tests for implementation. First, individual test problems, or test suites, which represent "industry standards," would be very useful for implementation. Effective industry-standard test problems are those that appear again and again in verification activities for a specific engineering or physics discipline. If they were to be carefully formulated, such test problems clearly would have great importance and should be used. Their present *ad hoc* role as a standard implies that there is insufficient clarity associated with (1) expected standards for comparison with these problems and (2) acceptance or success criteria when a comparison of the problem with a CFD code is performed. This issue is discussed below in relation to benchmarks.

In the absence of clear community standards for test problems, real or implied, a second approach is to choose test problems by consensus of the community

associated with the code, both developers and users. Expert opinion regarding discipline-specific test problems is important here. Developing consensus is not an easy job, especially if acceptance criteria for the test are likely to be publicly disseminated.

A third approach to choosing verification test elements is simply to construct specialized test problems that address specific needs arising in the structure of the test plan. Two examples of this approach are described below. In the first example, test problems are chosen to provide coverage of key code modules, paths through the software, or both. A second example is the use of so-called "Method of Manufactured Solutions."

The most accurate and traditional way to quantitatively measure the error in a computational solution is by comparing the computational solution to a highly accurate solution. However, highly accurate solutions are known only for a relatively small number of simplified problems. As categorized by the AIAA Guide [12], they are: analytical solutions, benchmark numerical solutions to the ODEs, and benchmark numerical solutions to the PDEs. The Method of Manufactured Solutions (discussed further below) is in the category of analytical solutions.

Analytical solutions are closed-form solutions to special cases of the PDEs that are represented in the conceptual model. These closed-form solutions are commonly represented by infinite series, complex integrals, and asymptotic expansions [110,137,151, 176,223,256,340,343]. Numerical methods are usually used to compute the infinite series, complex integrals, and asymptotic expansions in order to obtain the solutions of interest. However, the accuracy of these solutions can be quantified much more rigorously than can the accuracy of the numerical solutions of the conceptual model. The most significant practical shortcoming of analytical solutions is that they exist only for very simplified physics and geometries.

Consistency checks can also be usefully employed as benchmarks. A classic example is the use of intense point explosions in ideal gases to assess energy conservation in shock wave codes. Global checks on the conservation of appropriate quantities can be made [150], as well as checks that are related to the effect of the boundary conditions on the solution. One group of boundary condition tests is used to evaluate whether certain symmetry features are preserved in the solution. For example, if a plane of symmetry exists in the conceptual model, then the normal gradient of appropriate variables can be set to zero and a solution can be obtained. The same solution should also be obtained if this plane of symmetry condition is not imposed and the entire domain is solved. For unbounded domains, the boundaries of the computational domain are conceptually considered to be at infinity, i.e., they are infinitely far from the spatial region of interest. Typically, a user-defined parameter specifies how "far out" these boundaries are. If the boundaries are too close, the asymptotic conditions applied there may not be accurate. The usual method of determining the size of the computational domain is to systematically increase the domain until the solution is no longer dependent on the size of the domain that is compatible with the objectives of the computation. It is important to note that this exercise must be performed for a suitable grid and time step that are within the envelope of the grid and time-step convergence. A more comprehensive discussion of this topic is presented by Roache in his book [278].

When computational solutions are compared with highly accurate solutions, the comparisons should be examined along boundaries of interest or error norms should be computed over the entire solution domain. The accuracy of each of the dependent variables or functionals of interest should be determined. As previously mentioned, the required fidelity of the numerical solution varies greatly with the type of solution variable that is computed.

Benchmark ODE solutions are very accurate numerical solutions to special cases of the general PDEs. These ODEs commonly result from simplifying assumptions, such as simplified geometries and assumptions that result in the formation of similarity variables. Examples of benchmark ODE solutions in fluid dynamics are the Blasius solution for laminar flow over a flat plate, the Taylor–Maccoll solution for inviscid flow over a sharp cone, and the stagnation-region flow in two dimensions and three dimensions [151,256,340,343].

Benchmark PDE solutions are also very accurate numerical solutions to special cases of the PDEs or to special cases of the boundary conditions. Examples of benchmark PDE solutions in fluid dynamics are the following: incompressible laminar flow over a semi-infinite flat plate [87,329,347]; incompressible laminar flow over a parabolic plate [49,88,92]; incompressible laminar flow in a square cavity driven by a moving wall [34,130,131,141,147,228,298,318]; laminar natural convection in a square cavity [89,165], incompressible laminar flow over an infinite-length circular cylinder [28,33,177,218]; and incompressible laminar flow over a backward-facing step, with and without heat transfer [40,127,138,142].

The accuracy of the benchmark solutions clearly becomes more of an issue as one moves from benchmark ODE solutions to benchmark PDE solutions. In the literature, for example, one can find descriptions of computational simulations that are considered to be of high accuracy by the author but that are later found to be lacking. It is strongly recommended that no published solution be considered as a benchmark solution until the solution has been calculated very carefully by independent investigators, who preferably use different numerical approaches. We stress that code comparisons

are a verification activity—*NOT* a validation activity, regardless of what physical assumptions are made.

There are opportunities for applying additional ideas that are somewhat more methodical than defining algorithm tests. First, the "Method of Manufactured Solutions" (MMS) [278,289,307,320] is a method of custom-designing verification test problems of wide applicability. In this approach to constructing test problems, a specific form of the solution function is assumed to satisfy the PDE of interest. This function is inserted into the PDE, and all the derivatives are analytically computed manually or by using symbolic manipulation software such as MACSYMA™. The equation is rearranged such that all remaining terms in excess of the terms in the original PDE are grouped into a forcing-function or source term. This term is then considered to be simply added to the original PDE so that the assumed solution function satisfies the new PDE exactly. For example, in the Navier–Stokes equations this term can be considered to be a source term, i.e., a new term on the right-hand side of the PDE. The boundary conditions for the new PDE can be chosen to be the value of the solution function on the boundary (Dirichlet condition), a Neumann condition that can be analytically derived from the solution function, or a boundary condition of the third kind. This approach could be described as finding the problem, i.e., the PDE, for which we have assumed a solution.

Use of MMS in code verification activities requires that the computed source term and boundary conditions are programmed into the code and a numerical solution is computed. This procedure verifies, though for a narrow range of physical modeling, a large number of numerical aspects in the code, such as the numerical method, differencing technique, spatial transformation technique for grid generation, grid-spacing technique, and algorithm-coding correctness. Shih et al. [307] have applied this approach to the incompressible Navier–Stokes equations for laminar two-dimensional flow and have obtained an impressive exact solution to a variant of the classical lid-driven cavity problem for an arbitrary Reynolds number. It is highly recommended that incompressible Navier–Stokes codes be verified with this exact solution.

Salari and Knupp [289] have systematically applied MMS to the compressible and incompressible Navier–Stokes equations. They have also presented an interesting study of the "bug-finding" capability of the method by examining its performance on a set of 21 different coding mistakes to better understand what errors MMS is capable of resolving. MMS correctly diagnosed every error in the set that prevented the governing equations from being solved correctly. Salari and Knupp (see also [278]) noted that this method is incapable of detecting errors such as algorithm efficiency mistakes, where equations are still solved correctly by the algorithms, but where the coding is less efficient than optimal. These results suggest the desirability of coupling studies like Hatton's T1 experiments with a dynamic testing diagnosis like MMS to better correlate potential structural flaws with actual incorrect numerical performance. Such a study has not been performed.

MMS is an interesting blend of black-box testing and glass-box testing. We note that the method is actually code invasive, thus related to glass-box testing. MMS cannot be applied exclusively in black-box mode because of the need for specific coding intervention for each test that it creates, at least in the examples that we are aware of. Once that coding is accomplished, however, the gathering of results proceeds in a fully black-box manner. There does not appear to be any analog of this testing methodology in the work of Beizer [36] and Kaner et al. [175], which is rather surprising.

The third pressing issue for algorithm testing is quantitative assessment of code performance on the chosen test problems. A clear assessment of code performance on algorithm test problems is crucial. To develop such an assessment requires precise definition of how code calculations should be compared with the fiducial. It also requires the most quantitative possible definition of acceptance criteria. If a clear definition of acceptable code performance on the test cannot be assigned, then a clear definition of unacceptable performance should be described instead. We acknowledge that the construction of acceptance criteria may be very difficult. Yet, it is hard to fathom what benefit results from running test problems in which criteria for success or failure cannot be stated. The suggested structure for verification problems given in [12] recognizes the complexity of this issue by attempting to limit the complexity of comparison and increasing the clarity of determining success or failure in the recommended benchmark definitions. The advantage of simple benchmark problems is that it is relatively easy to measure how accurately a code executes the problem, although even for problems with analytical solutions like shock tube problems, quantitative acceptance may be challenging. The disadvantage of simple benchmarks is that they tend to be remote from the applications of interest, especially for complex codes. As a general principle, our view is that benchmarks that are similar to intended applications tend to be oxymorons. The closer "benchmarks" are to the intended application, the less likely it is that a correct baseline solution for them will be known for the problem.

We have just suggested the special weight that "benchmark" problems play in algorithm test plans. There are two senses in which we mean benchmarks for use in algorithm testing. *Weak sense benchmarks* are test problems that are in common ad hoc use by various algorithm developers or code projects for the purpose of

assessing numerical accuracy. These were illustrated in the above discussion.

We define *strong sense benchmarks* fundamentally to be engineering *standards*. Test problems that are strong sense benchmarks have their definition, use, and impact precisely defined and formally documented, typically by professional societies, academic institutions, or nonprofit organizations.

More specifically, it is our view that strong sense benchmarks are defined by the following four factors:

- Exact, standardized, frozen, and promulgated definition of the benchmark.
- Exact, standardized, and promulgated statement of the purpose of the benchmark. This addresses its role and application in a comprehensive test plan for a code, for example.
- Exact, standardized, frozen, and promulgated requirements for comparison of codes with the benchmark results.
- Exact, standardized, frozen, and promulgated definition of acceptance criteria for comparison of codes with the benchmark results. The criteria can be phrased either in terms of success or in terms of failure.

Do strong sense benchmarks exist? We believe that the answer is "certainly not in CFD." In the long run, though, there appear to be two (possibly more) efforts that may move in the direction of establishing strong sense benchmarks. One effort is the NPARC Alliance [314], which is attempting to standardize the definition and purpose of verification and validation problems in CFD. Their work is concentrated on the establishment and acceptance of an archive for accepted verification and validation problems by the CFD community. This effort builds on consensus in the CFD professional community regarding important verification and validation problems.

The ERCOFTAC Special Interest Group on "Quality and Trust in Industrial CFD" is an effort that goes one step further. This European group has begun to promulgate best practice guidelines [58], including benchmarks that may eventually become "strong sense" in our meaning. ERCOFTAC and the associated working group QNET-CFD are discussed in the recent paper by Hutton and Casey [166].

While it is too early to judge exactly where these efforts are going, we have noted that none of these groups have dealt with the most sensitive issue in our definition of strong sense benchmarks, that is, the issue of establishing stringent standards for comparison with benchmarks and measuring success. It is very clear to us that establishing these particular standards, as well as the others in our definition, will be a very difficult task in the current climate in which CFD is pursued. The realities of competition between commercially available CFD codes, competition between organizations that rely on CFD codes, and competition between countries may stifle this endeavor. We believe the maturation of CFD will suffer if strong sense benchmarks are not developed.

A stimulus to the development of benchmarks closer to our strong sense meaning is the increasingly aggressive posture of certain professional journals regarding minimal standards for the computational work that they publish. For example, see [274] for a discussion of issues associated with the development of editorial policy dealing with the reporting of numerical accuracy in the ASME *Journal of Fluids Engineering* [280]. Currently, journals that have editorial policies emphasize the assessment of numerical accuracy for the solutions presented in papers submitted for publication. This is actually a weaker statement of verification than our weak sense benchmark statements. Evolution of the stance of the professional journals would require the eventual use of something like strong sense benchmarks as further characterization and demonstration of computational quality. Based on the controversies that erupted over editorial requirements on the reporting of numerical accuracy, whether or not strong sense benchmarks will actually be used in refereed journals for the foreseeable future is obviously open to question. The decision is ultimately one to be made by the professional CFD community.

3.4.4. Software quality engineering

We conclude by providing only a brief introduction to some literature in SQE that is relevant to the overall problem of verification assessment. A number of modern texts are available that describe current practice in the SQE field [23,26,84,91,107,113,173,200,283]. Much of the research and development in SQE procedures has been fueled by computer-controlled systems that require extremely reliable and secure software, as in Johnson's avionic systems application referenced previously. Such systems are commonly referred to as "high-integrity systems". Examples of high-integrity systems other than avionics systems are control systems for nuclear power reactors and software for nuclear weapon security and safety. The scientific software community has much to learn from SQE procedures that have been developed for these systems as high-consequence scientific computing becomes more prevalent.

Standards also play an important role in the impact of SQE. Paulk et al. [260] describe the Capability Maturity Model (CMM), an important standard in the current Department of Defense software acquisition process. Sanders and Curran [292] describe elements of the ISO 9000 standards. A criticism of some important elements of the CMM is found in [233].

The SQE field typically uses definitions of verification and validation that have been developed by the IEEE [61]. When consulting references in the SQE field, the

interested reader should remember the differences in terminology between those used here and those developed by the IEEE.

Various authors in the SQE community follow different approaches to the process of code verification [23,26,84,91,115,186,200,203,283,315,339]. As an example, we outline the approach taken by Wallace et al. [339] and Marciniak [203]. In this approach, software verification activities can be generally classified into management activities and technical activities; the major management and technical activities are listed in Table 1. Because each of the management and technical activities are discussed at length in [339], details of the activities are not presented here.

4. Validation assessment

4.1. Fundamentals of validation

4.1.1. Validation and prediction

The fundamental concept of "Comparison and Test of Agreement" between the computational results and experimental data depicted in Fig. 3, presented pre-

Table 1
Major software verification activities [339]

Activity	Tasks
Software verification management	Planning Monitoring Evaluating results, impact of change Reporting
Software requirements verification	Review of concept documentation Traceability Software requirements evaluation Interface analysis Initial planning for software system test Reporting
Software design verification	Traceability analysis Software design evaluation Interface analysis Initial planning for unit test Initial planning for software integration test Reporting
Code verification	Traceability analysis Code evaluation Interface analysis Completion of unit test preparation Reporting
Unit test	Unit test execution Reporting
Software integration test	Completion of software integration test preparation Execution of software integration tests Reporting
Software system test	Completion of software system test preparation Execution of software system tests Reporting
Software installation test	Installation configuration audit Reporting
Software operation and maintenance	Impact of change analysis Repeat management activities Repeat technical activities

viously, is clearly consistent with the view of validation held by engineers and scientists. However, we believe the reason validation science is so difficult, even with years of effort by a large number of researchers in diverse fields, can be captured in the following two questions: "How is validation executed?" and "What do model validation activities imply?" Concerning the first question, the approach to validation must be analogous to the strategy of verification discussed in Section 3.1; that is, validation of a model or code cannot be mathematically proven; validation can only be assessed for individual realizations of nature. Individual computational outcomes of a model are validated; codes are not validated [278,279]. This fundamental concept has been resolved philosophically [56,263,264], but commonly in the CFD literature a favorable comparison of computational results and experimental data motivates the author (code developer, code user, or code salesperson) to declare the entire computer model is "validated". This view is not defensible in any meaningful way for complex physical processes. We believe the more difficult of the two questions to answer is the second, which can be stated similarly as "What can be inferred when we make a prediction of some physical process with a model that has completed some level of validation?" This question has not received much attention in the literature, probably because of the diverse views toward the meaning and processes of validation.

A significant step forward in clarifying the second question and segregating it from validation was taken in the AIAA Guide [12]. The AIAA Guide recommends that the meaning of prediction must be viewed in the context of the validation database. In the AIAA Guide, prediction is defined as "use of a computational model to foretell the state of a physical system under conditions for which the computational model has not been validated". A prediction refers to the computational simulation of a specific case of interest that is *different* from cases that have been validated. This definition of prediction is more restrictive than the general scientific meaning of prediction because it eliminates past comparisons of computational results with experimental data. This definition segregates the general meaning of prediction and only refers to prediction, not retrodiction (replication of previously obtained results). If this restriction is not made, then one is only demonstrating previous agreement with experimental data in the validation database. The results of the process of validation should be viewed as historical statements. Thus, the validation database represents reproducible evidence that a model has achieved a given level of accuracy in the solution of specified problems. From this perspective, it becomes clear that validation comparisons do not directly make claims about the accuracy of predictions; they allow inferences. We suggest the relationship between validation and prediction can be viewed as shown in Fig. 7.

Fig. 7 attempts to capture the distinction between validation and prediction. The bottom portion of the figure represents the validation process. Although it is not readily apparent, the validation process in Fig. 7 is fundamentally the same as in Fig. 3. In Fig. 7, the block "Validation Experiments" produces one or more realizations of the "Real World". The "Experimental Outcomes" are the physical realizations, i.e., the experimental data from the experiment. The physical conditions from the actual validation experiments, i.e, model input parameters, initial conditions, and boundary conditions, are input to the "Computational Model." The computational model produces the "Computational Results of Experimental Outcomes." These

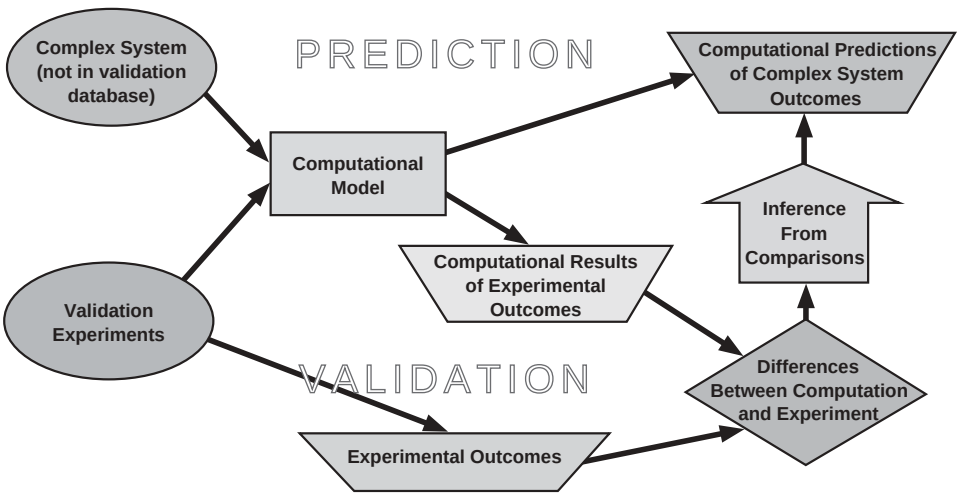

Fig. 7. Relationship of validation to prediction.

results are then compared with the experimentally determined outcomes in the block "Differences Between Computation and Experiment." Based on the magnitude of these differences in quantities of interest in the simulation, and based on the understanding of the physical process, an "Inference from Comparisons" is made.

The upper portion of Fig. 7 represents the prediction process. The "Complex System" of interest drives the entire modeling and simulation process, but the vast majority of the realizations of interest, i. e., predictions, are not in the validation database. That is, when a physical realization is conducted as part of the validation database, regardless of the tier as discussed in Section 2.3: Methodology for Verification and Validation, then the realization becomes part of the "Validation Experiments." Predictions for conditions of interest are made using the "Computational Model" which result in "Computational Predictions of Complex System Outcomes." The confidence in these predictions is determined by the "Inference from Comparisons" and the level of understanding of the physical process.

The logical inference of a computational model and its validation database is analogous to classical scientific theories. However, we argue that the strength or confidence in the inference from computational simulation is, and should be, much weaker than traditional scientific theories. Computational simulation relies on the same logic as traditional science, but it also relies on many additional theoretical issues, e.g., discretization algorithms and grid quality, and practical issues, e.g., computer hardware, operating system software, source code reliability, and analyst skill, that are not present in classical science. One of the key classical theoretical issues is the state of knowledge of the process being modeled, i.e., the complexity of the process. For physical processes that are well understood both physically and mathematically, the inference can be quite strong. An example is laminar, incompressible, single-phase, single-fluid, nonheat-conducting, nonreacting, wall-bounded Newtonian flow. For a stationary geometry, i.e., fixed boundary conditions, there is only one nondimensional parameter remaining in the mathematical formulation: the Reynolds number. Even for this simple flow, however, as the Reynolds number or geometry is changed, the flow can undergo unanticipated changes in character. Examples are: change from two-dimensional to three-dimensional flow, and change from a steady flow to unsteady flow. The emphasis here is that the strength of the inference from the validation database becomes weaker as the complexity of the process increases. A general mathematical method for determining how the inference value degrades as the physical process becomes more complex has not been formulated. For example, in a complex physical process how do you determine "how nearby" is the prediction case from cases in the validation database? Struggling with the strength or quantification of the inference in a prediction is presently an important topic of research [66].

Returning to the first question of how validation is executed, we believe this question implies three related issues:

- How is a validation comparison quantitatively measured?
- What is a productive methodology for the determination of needed validation experiments?
- What are the characteristics of a validation experiment?

To properly deal with the first issue requires a more careful distinction between uncertainty and error than has traditionally been made in the literature. Our meaning of error, both acknowledged and unacknowledged, was discussed in Section 3.2. The term "uncertainty," in its technical sense, seems to have two rather different meanings. The first meaning of uncertainty has its roots in statistics: the estimated amount or percentage by which an observed or calculated value may differ from the true value. This meaning of uncertainty has proven its usefulness over many decades, particularly in statistical estimation of experimental measurement uncertainty [75]. The second meaning of uncertainty is connected to issues of nondeterminism, particularly with regard to the prediction of future events and the estimation of the reliability of systems. The probabilistic risk, safety, and performance assessment communities, as well as the reliability engineering community, use the term "uncertainty" in this latter sense. When dealing with the topic of validation, it is clear that both meanings are appropriate and useful.

Concerning the nondeterminism meaning of uncertainty, the risk assessment community (see, for example, [22,83,114,122,124,154,155,161,164,244,257,259,267,284]) and the information theory community (see, for example, [16,105,140,184,185,187,303,316] have concluded that uncertainty should be divided into two types: aleatory uncertainty and epistemic uncertainty. Aleatory uncertainty is used to describe the inherent variation associated with the physical system or the environment under consideration. Sources of aleatory uncertainty can commonly be singled out from other contributors to uncertainty by their representation as randomly distributed quantities that can take on values in an established or known range, but for which the exact value will vary by chance from unit to unit or from time to time. The mathematical representation most commonly used for aleatory uncertainty is a probability distribution. Aleatory uncertainty is also referred to in the literature as variability, irreducible uncertainty, inherent uncertainty, and stochastic uncertainty.

Epistemic uncertainty as a source of nondeterministic behavior derives from some level of ignorance or lack of knowledge of the system or the environment. As a result, an increase in knowledge or information can lead to a reduction in the predicted uncertainty of the response of the system, all things being equal. Examples of sources of epistemic uncertainty are when there is little or no experimental data for a fixed (but unknown) physical parameter, limited understanding of complex physical processes, and little knowledge of an inflow or outflow boundary condition in an experiment. Epistemic uncertainty is also referred to in the literature as reducible uncertainty, subjective uncertainty, and model form uncertainty.

Although further discussion of the distinction between aleatory and epistemic uncertainty is beyond the scope of this paper, we believe the CFD community should be made aware of related developments in other fields. Concerning this paper, when we use the term "uncertainty" we will imply both types of uncertainty.

4.1.2. Validation error and uncertainty

We now describe the fundamental issues concerning how a validation comparison is quantitatively measured. Let u be an arbitrary system response measure that is both computed and experimentally measured. Let Δ be the difference between the true or exact value of nature, u_{nature}, and the computational result, u_{discrete}, so that

$$\Delta = (u_{\text{nature}} - u_{\text{discrete}}). \tag{12}$$

This can be rewritten as

$$\Delta = (u_{\text{nature}} - u_{\text{exp}}) + (u_{\text{exp}} - u_{\text{exact}}) + (u_{\text{exact}} - u_{\text{discrete}}), \tag{13}$$

where u_{exp} is the value measured in the experiment and u_{exact} is the exact solution of the continuum PDEs and the given set of initial conditions and boundary conditions of the mathematical model. Using Eq. (2) to expand the last term, and rewriting Eq. (13), one has

$$\Delta = E_1 + E_2 + E_3 + E_4, \tag{14}$$

where

$$E_1 = (u_{\text{nature}} - u_{\text{exp}}),$$

$$E_2 = (u_{\text{exp}} - u_{\text{exact}}),$$

$$E_3 = (u_{\text{exact}} - u_{h,\tau\to 0}),$$

$$E_4 = (u_{h,\tau\to 0} - u_{h,\tau,I,C}).$$

E_1 through E_4 conceptually represent all of the sources of uncertainty and error in a comparison of computational results and experimental data. E_1 and E_2 will be discussed here, E_3 and E_4 were discussed in Section 3.2.

E_1 represents all errors due to measurement of a physical quantity. In one sense, u_{nature} could be viewed as a deterministic quantity; that is, given a fixed set of physical conditions, the same result for u_{nature} will be realized. (In this discussion we ignore bifurcation and chaotic phenomena.) However, it is more realistic and practical to consider u_{nature} as a random variable because exactly the same physical conditions do not exist from one physical realization to the next. For example, it is common in most wind tunnel experiments that inflow and outflow boundary conditions are well controlled, but they are not precisely deterministic. In free-flight testing, for example, atmospheric conditions are uncontrolled, but it is important to characterize as best as possible the temporal and spatial conditions of the atmosphere. u_{exp} is clearly a random variable because of random (precision) uncertainty in the experimental measurement. Then considering E_1 as a random variable, it can be seen that E_1 can be nonzero due to both a shift in the mean of the distribution, i.e., a shift due to bias error, and a difference in the random distribution exhibited by nature as compared to that measured in the experiment. We believe the more important error of these causes is the bias (systematic) error in an experimental measurement. In fluid-dynamic measurements, examples are temperature drift of a pressure transducer, error in an amplifier gain, error in a free-stream Mach number, error in data reduction software, and intrusive flow measurements that alter the measured flow.

E_2 represents all errors and uncertainties resulting from differences between the experimental measurement and the exact solution of the continuum PDEs that are attempting to describe the experimental event. Note that u_{exact} also contains all of the initial conditions, boundary conditions and physical parameters that are needed to model the experiment. The term E_2 is usually referred to as "modeling error," but based on the definitions discussed here, it is seen that E_2 can involve both error and uncertainty. The most common example of modeling errors consists of those errors that are acknowledged in the modeling of well-understood approximations to the physical process. Examples of such errors are assumption of a continuum fluid, assumption of incompressibility in low-speed flow, assumption of a constant Prandtl number in turbulent flow, and assumption of turbulence models. Unacknowledged errors can also cause E_2 to be nonzero, for example, mistakes in the input parameters describing the thermodynamic properties of the fluid or the geometry of the experiment. Examples of uncertainties in modeling are poorly known reaction-rate parameters in chemically reacting turbulent flow, certain parameters in multiphase flows, surface roughness in turbulent flow, and parameters required for the computational simulation that were not measured in the experiment. Sometimes it is debatable whether a quantity should be considered as an acknowledged error or as an uncertainty in modeling. If alternate plausible models of a physical process can be clearly

ordered according to their accuracy in representing the physical process, then we believe the modeling assumption should be viewed as an acknowledged error. If the models cannot be so ordered, e.g., they are all within the same class of models, or the physical process is poorly understood, then we believe it is more constructive to view the contribution to E_2 as an uncertainty. Recent work [243] has attempted to identify the dominant sources of error and uncertainty in all phases of the modeling and simulation process.

We close this subsection with three points of emphasis on Eq. (14). First, the summation of the four terms clearly suggests that even if Δ was zero for a given comparison of computational results and experimental data, one cannot argue that all the terms are identically zero. An error or uncertainty in any of these terms can cancel the error and uncertainty in any other term, or combination of terms. To reduce the likelihood of this occurring, verification activities and solution-accuracy assessment activities are meant to provide estimates of E_3 and E_4. Validation activities attempt to provide an estimate of the magnitude of the experimental uncertainty, E_1, so that one can best estimate the magnitude of the modeling error and uncertainty, E_2. Stated differently, validation comparisons are primarily meant to evaluate the fidelity of the continuum mathematics model of the physical process, not to find code verification errors or solution accuracy errors. This emphasizes the importance of robust verification activities *before* validation activities are conducted.

Second, for most simulations of complex physics or multidisciplinary engineering systems, Δ is not small—possibly far from small. These simulations commonly involve important physical modeling parameters that are not known a priori or that are averaged or effective parameters. For complex flows, in fact, some of the parameters embodied in u_{exact} are considered to be adjustable or tunable parameters; that is, they are adjusted so that the sum of the four terms is zero ($\Delta = 0$). For example, in turbulence models or reacting flow models it is accepted practice to calibrate or adjust modeling parameters to minimize the error over a range of experiments. This practice is fully defensible when it has been demonstrated that E_3 and E_4 are small. Another practical engineering situation occurs when grid-resolved solutions cannot be achieved because the computer resources needed for the simulation are not available, that is, E_4 is known to be large. Then it is common engineering practice to adjust the modeling parameters so that improved agreement with the experimental data is obtained. This type of calibration is expedient in certain engineering situations, but it is weakly defensible.

When these types of practical engineering activities occur, the term *calibration* more appropriately describes the process than does validation [12]. In other words, calibration, or parameter estimation, is a response to obtain needed agreement between the computational results and the experimental data. Although calibration of models is required for essentially all complex processes, this paper does not specifically deal with this topic. However, we believe it is important to clearly call attention to the difference between validation and calibration.

Third, because of the nondeterministic, i.e., stochastic, nature of E_1 and E_2, Δ must also be considered as a random variable. As a result, the most appropriate manner in which to interpret Eq. (14) is that Δ is the sum of the errors and uncertainties represented by the probability distributions for E_1 and E_2, as well as the errors contributed by E_3 and E_4. The nondeterministic nature of Δ, E_1 and E_2 will be discussed in Sections 4.4, 4.5, and 4.7. The bias error feature of E_1 and E_2 will be discussed in Sections 4.4 and 4.6.

4.2. Construction of a validation experimental hierarchy

4.2.1. Hierarchy strategy

As discussed in Section 2.3, the validation hierarchy recommended by the AIAA Guide is constructed by beginning with the complete system of interest. This hierarchical view of validation is distinctly an engineering perspective, as opposed to a scientific or research perspective. The purpose of the validation hierarchy is to help identify a range of experiments, possible separation of coupled physics, and levels of complexity, all of which are related to an engineering system, so that computer codes from different disciplines can be evaluated. Stated differently, the validation hierarchy must be application driven to be of engineering value, not code driven. As one constructs each lower tier, the emphasis moves from multiple coupled codes simulating different types of physics to single codes simulating a particular type of physics. In parallel with the simplification process, the focus on the actual operating conditions of the complete system should not be lost. The construction of these hierarchical tiers and the identification of the types of experiments that should be conducted at each tier are formidable challenges. There are many ways of constructing the tiers; no single construction is best for all cases. We would draw the analogy of constructing validation hierarchies to the construction of control volumes in fluid-dynamic analyses. Many varieties of control volumes can be drawn; some lead nowhere, and some are very useful for the task at hand. The construction should emphasize the modeling and simulation capability that is desired to be validated, whether it is CFD or other computational disciplines. Analogous tier structures can be developed for structural dynamics and electrodynamics, for example, when the engineering system of interest involves these disciplines.

A good hierarchical tier construction is one that accomplishes two tasks. First, the construction carefully disassembles the complete system into tiers in which each lower-level tier has one less level of physical complexity. For complex engineered systems, this may require more than the three building-block tiers shown in Fig. 2, presented previously. The types of physical complexity that could be uncoupled from one tier to the next are spatial dimensionality, temporal nature, geometric complexity, and physical-process coupling. The most important of these types to decouple or segregate into separate effects experiments, from one tier to the next, is physical-process coupling, which commonly contains the highest nonlinearity of the various contributors. It is important to recognize the nonlinear nature of all of the contributors in the construction of the tiers because the philosophy of the tier construction rests heavily on linear-system thinking. That is, confidence in the computational capability for the complete system can be built from assessment of computational capability of each of its parts. The complete systems of interest clearly do not have to be linear, but the philosophy of the hierarchical validation approach loses some of its utility and strength for strong nonlinear coupling from one tier to the next.

The second task accomplished by a good hierarchical tier construction is the selection of individual experiments in a tier that are practically attainable and able to produce validation-quality data. In other words, the individual experiments should be physically achievable given the experimental test facilities, budget, and schedule, and they should be capable of producing quantitative experimental measurements of multiple system-response measures that can test the code. As discussed in Section 2.3, the ability to conduct a true validation experiment at the complete system tier is extremely difficult, if not impossible, for complex systems. At the subsystem tier, it is usually feasible to conduct validation experiments, but it is still quite difficult and expensive. One usually chooses a single hardware subsystem or group of subsystems that are closely related in terms of physical processes or functionality. For complex subsystems, one might want to add a new tier below subsystems called components. As with subsystems, this tier would consist of actual operational hardware components. When one defines the individual experiments at the benchmark-tier level, then special hardware, i.e., nonoperational, nonfunctional hardware must be fabricated. The benchmark tier is probably the most difficult to construct because it represents the transition from a hardware focus in the two top tiers to a physics-based focus in the bottom tiers of the hierarchy. At the bottom tier, unit problems, one should identify simple geometry experiments that have a single element of physical-process complexity. As with the subsystem tier, an additional tier may need to be added to attain only one element of physics at the bottom tier. Also, the experiment must be highly characterized so that it can provide the necessary data to the computational code, and the experiment must be conducted so that experimental uncertainty can be estimated precisely. As discussed in Section 2.3, high-quality validation experiments are practically attainable at the benchmark and unit-problem tiers, but usually not at the system or subsystem tiers for complex systems.

4.2.2. Hierarchy example

When the concept of a validation hierarchy was formally proposed [310], the application was for impeller/diffuser interaction on rotating machinery. Recently, a hierarchical tier structure was constructed for a complex, multidisciplinary system: an air-launched, air-breathing, hypersonic cruise missile [246]. Following this example, assume the missile has an autonomous guidance, navigation, and control (GNC) system, an on-board optical target seeker, and a warhead. Fig. 8 shows a hierarchical validation structure for the hypersonic cruise missile that forms the basis of this discussion. We refer to the missile as the complete system and the following as systems: propulsion, airframe, GNC, and warhead. These systems would normally be expected in the engineering design of such a vehicle; however, additional elements could be added or the named elements could be subdivided to emphasize systems of importance to computational analysts from different disciplines. The launch aircraft is not included at the system level because its location in the hierarchy would be at the next higher level, i.e., above the cruise missile. The structure shown is not unique and it is not necessarily optimum. In addition, the structure shown emphasizes the aero/thermal protection subsystem, as will be discussed shortly.

At the subsystem tier, we have identified the following elements: aero/thermal protection, structural, and electrodynamics. Electrodynamics deals with the computational simulation of radio-frequency detectability of the cruise missile, e.g., radar cross-section at different viewing angles of the missile. Only three elements are identified at the subsystem tier because they are the primary engineering design features that deal with the airframe. Arrows drawn from the system-tier elements to the subsystem-tier elements indicate the primary elements that influence the lower tier. Recall at the subsystem tier that each element should be identified with functional hardware of the cruise missile. Notice, however, how one would begin to conduct validation experiments at this tier depending on the computational discipline of interest. For example, the aero/thermal subsystem would contain the actual thermal protective coating over the metal skin of the missile, the actual metallic skin of the vehicle, much of the substructure

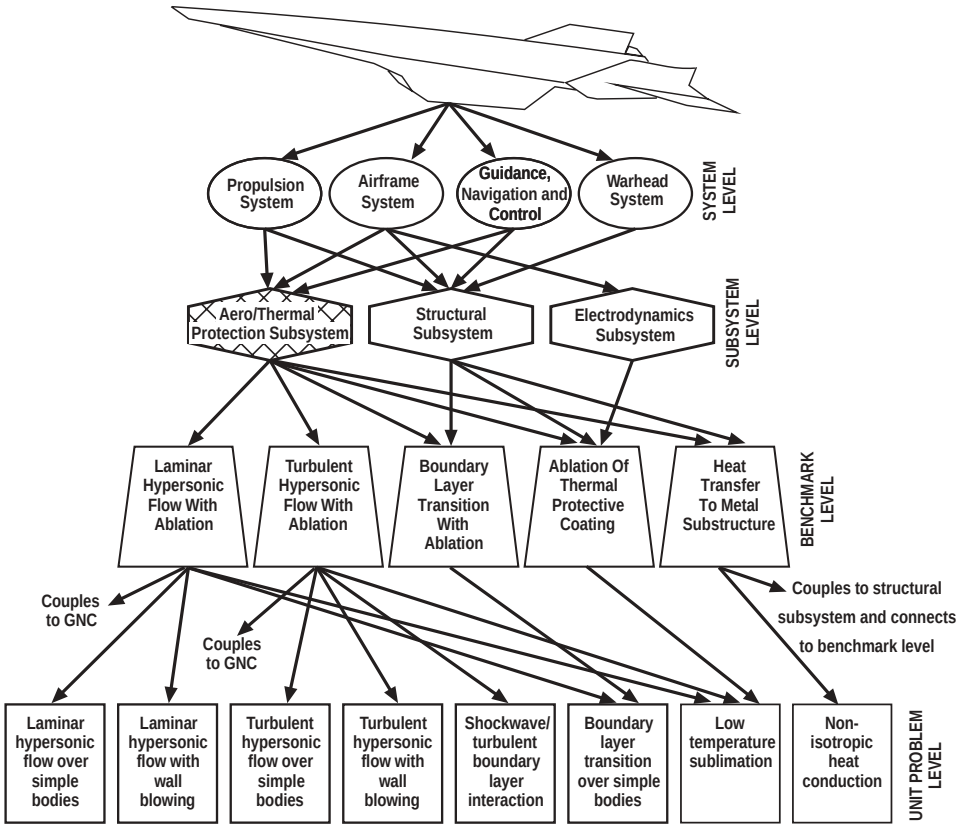

Fig. 8. Validation hierarchy for a hypersonic cruise missile [246].

under the skin of the vehicle, all of the functional lifting and control surfaces, and the internal flow path for the propulsion system. However, the aero/thermal subsystem probably would not contain any other hardware inside the vehicle, unless some particular heat conduction path was critical. If one was interested in validation experiments for a structural dynamics code, then the structural subsystem identified would be quite different from the aero/thermal subsystem. For example, the structural subsystem would contain essentially every piece of hardware from the missile because every part of the structure is mechanically coupled with every other part of the structure. Structural modes are influenced by all mechanically connected hardware, some to a lesser extent than others. Certain simplifications of the hardware, however, would be appropriate. For example, one could substitute mass-mockups for certain systems, such as the warhead and the completely functional propulsion system, with little loss in fidelity. However, the structural excitation modes caused by the propulsion system would be quite important in the validation of the structural subsystem.

At the benchmark tier, the following elements are identified: laminar hypersonic flow with ablation, turbulent hypersonic flow with ablation, boundary-layer transition with ablation, ablation of thermal protective coating, and heat transfer to metal substructure. In Fig. 8 we only show examples of key elements at the benchmark tier that are related to the aerothermal protection subsystem. We do not show any of the benchmark tier elements that would be derived from the structural or electrodynamics subsystems. The arrows drawn from these two subsystems to the benchmark level only show coupling from these two subsystems to elements at the benchmark tier that are derived from the aerothermal protection subsystem. At the benchmark tier one fabricates specialized, nonfunctional hardware. For example, the laminar, turbulent, and boundary-layer-transition elements may not contain the actual ablative coating of the missile. Instead, a simpler material might be used—one that would produce wall blowing and possibly gases or particles that may react within the boundary layer, but yet simpler than the typically complex gas and particle chemistry that results from actual ablative materials. The arrow from the structural subsystem to the boundary-layer-transition element is drawn to show that structural vibration modes of the surface can influence transition. The

element for ablation of the thermal protective coating may use the actual material on the missile, but the validation experiment may be conducted, for example, at conditions that are attainable in arc-jet tunnels. An additional arrow is drawn from each of the elements for hypersonic flow with ablation that are marked "Couples to GNC." These arrows indicate a significant coupling of the flow field to the optical seeker in the GNC hierarchy (not shown here). The element for the heat-transfer-to-metal substructure shows an arrow that would connect to elements at the benchmark level in the structural subsystem hierarchical tree. This arrow indicates the coupling that will result in thermally induced stresses and cause temperature-dependent material properties to be considered in the structural simulation.

At the unit-problem tier, the following elements are identified: laminar hypersonic flow over simple bodies, laminar hypersonic flow with wall blowing, turbulent hypersonic flow over simple bodies, turbulent hypersonic flow with wall blowing, shock wave/turbulent boundary layer interaction, boundary layer transition over simple bodies, low temperature sublimation, and nonisotropic heat conduction. Many other elements could be identified at this tier, but these are representative of the types of validation experiments that would be conducted at the unit-problem tier. The identification of elements at this tier is easier than at the benchmark tier because unit-problem elements are more closely related to traditional, physical-discovery or mathematical-model-building experiments in fluid dynamics and heat transfer.

A point of clarification should be made concerning experiments at the lower levels of the hierarchy, particularly at the unit-problem level. Some have referred to experiments, such as laminar hypersonic flow in a wind tunnel, as a "simulation" of the flight vehicle in the atmosphere. From the perspective of a project engineer interested in performance of the "real" vehicle, this view about experimentation is appropriate. From the perspective of one conducting a validation experiment, however, this view only confuses the issue. That is, an experiment conducted at any level is a physical realization of a process whose results can be compared to a computational simulation of the actual physical experiment conducted. The relationship of the physical experiment to some engineering system is not the critical issue with regard to the "reality" of the validation experiment. Any experiment is reality that can be of value in validation, but this view is not commonly appreciated by the project engineer who is interested in performance of the "real" vehicle.

Even after a validation hierarchical structure like that illustrated in Fig. 8 has been constructed, another practical issue remains: identifying which validation experiments are the most important and, as a result, should be conducted first, if possible. Useful information for prioritizing the validation experiments can be found in a procedure used to assess the safety of nuclear power reactors. This reactor-safety procedure, referred to as the Phenomena Identification Ranking Table (PIRT), was developed for prioritizing which physical phenomena are the most important to analyze and understand [344]. The PIRT focuses attention on the application of the code to the operating conditions of interest for the complete system. Although the PIRT has not been used in the aerospace industry, we believe this procedure can be used in conjunction with the present hierarchical structure as an aid in prioritizing validation experiments.

To better explain how the validation hierarchy of the airframe system is related to the validation hierarchy of the propulsion, GNC, and warhead systems, we refer to Fig. 9 [246]. The validation hierarchy of each of these four systems could be viewed as the primary facets of a four-sided pyramid. The airframe facet was divided into three additional facets, each representing the three subsystems: aero/thermal protection, structural, and electrodynamics. The propulsion system could be divided into four additional facets to represent its subsystems: compressor, combustor, turbine, and thermal signature. The GNC and the warhead systems could be divided into subsystems appropriate to each. On the surface of this multifaceted pyramid, one could more clearly and easily indicate the coupling from one facet to another. For example, we discussed the coupling of laminar and hypersonic flow with ablation to the optical seeker of the GNC system. This coupling would be shown by an arrow connecting these hypersonic flow elements to appropriate elements on the GNC facet of the pyramid.

The validation pyramid stresses the system-engineering viewpoint, as opposed to a specific discipline

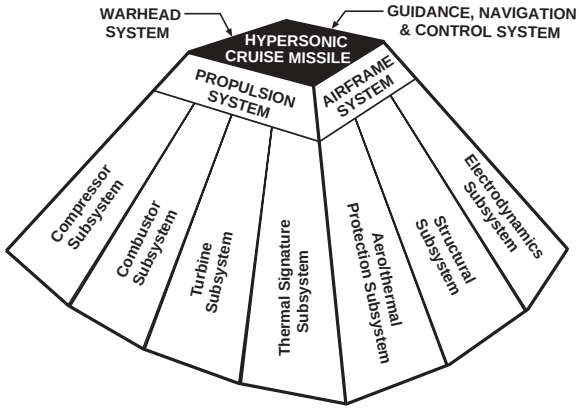

Fig. 9. Validation pyramid for a hypersonic cruise missile [246].

viewpoint, in modeling-and-simulation-based design. Each facet of the pyramid can then be devoted to identifying validation experiments for each computational code responsible for part of the design of the system. As one traverses around the top of the pyramid, the number of facets is equal to the number of systems that are identified. As one traverses around the bottom of the pyramid, the number of facets is equal to the total number of major computer codes used in the analysis of the engineering system, i.e., the number of codes that require validation activities for the intended application. For the example of the hypersonic cruise missile, if the code that simulates surface ablation is a separate code from the aerodynamics code, then an additional facet on the pyramid is added on the aero/thermal subsystem facet. We strongly believe this type of system-level thinking is necessary to increase the confidence in complex systems that are designed, manufactured, and deployed with reduced levels of testing.

Two final comments are in order concerning the construction of a validation hierarchy. First, the location of a particular validation experiment within the hierarchy must be determined relative to the complete system of interest, i.e., it must be appropriately related to all of the experiments above it, below it, and in the same tier. Stated differently, the same validation experiment can be at different tiers for validation hierarchies that are constructed for different complex systems of interest. For example, the same turbulent-separated-flow experiment could be at the unit-problem tier in a complex system and at the benchmark tier in a simpler engineering system. Second, a validation hierarchy is constructed for a particular engineered system operating under a particular class of operating conditions, for example, normal operating conditions. A new hierarchical pyramid would be constructed if one were interested in computationally analyzing other classes of system operating conditions. For example, if one were interested in failure or crash scenarios of the system, then one would construct a different pyramid because different modeling and simulation codes would come into play. Similarly, if the system would have to function under hostile conditions, e.g., under physical or electromagnetic attack, then a different pyramid would also be constructed.

4.3. Guidelines for validation experiments

Many researchers, analysts, and managers ask the question: "What is a validation experiment?" or "How is a validation experiment different from other experiments?" These are appropriate questions. We suggest that traditional experiments could generally be grouped into three categories. The first category comprises experiments that are conducted primarily for the purpose of improving the fundamental understanding of some physical process. Sometimes these are referred to as physical-discovery experiments. Examples are experiments that measure fundamental turbulence characteristics, detailed reaction chemistry in combustion experiments, and experiments for flows in thermochemical nonequilibrium. The second category of traditional experiments consists of those conducted primarily for constructing or improving mathematical models of fairly well-understood flows. Examples are experiments to (1) measure reaction-rate parameters in reacting flows, (2) determine turbulence modeling parameters best suited for separated flows, and (3) determine thermal emissivity of fluid-borne particles or surfaces. The third category of traditional experiments includes those that determine or improve the reliability, performance, or safety of components, subsystems, or complete systems. These experiments are commonly called "tests" of engineered components or systems. Examples are tests of new combustor designs, compressors, turbopumps, gas turbine engines, and rocket engines. We would also include in this last category traditional wind-tunnel tests conducted for the purpose of measuring vehicle and control-surface forces and moments.

We argue that validation experiments constitute a new type of experiment. A validation experiment is conducted for the primary purpose of determining the validity, or predictive accuracy, of a computational modeling and simulation capability. In other words, a validation experiment is designed, executed, and analyzed for the purpose of quantitatively determining the ability of a mathematical model and its embodiment in a computer code to simulate a well-characterized physical process. Thus, in a validation experiment "the code is the customer" or, if you like, "the computational analyst is the customer." Only during the last 10–20 years has computational simulation matured to the point where it could even be considered as a customer. As modern technology increasingly moves toward engineering systems that are designed, and possibly even fielded, based on modeling and simulation, then modeling and simulation itself will increasingly become the customer of experiments.

As mentioned in Section 2.2, a number of researchers, particularly experimentalists, have been slowly developing the concepts of a validation experiment. During the past several years, a group of researchers at Sandia National Laboratories has been developing philosophical guidelines and procedures for designing and conducting a validation experiment. Although the following six guidelines and procedures were developed in a joint computational and experimental program conducted in a wind tunnel, they apply over the entire range of fluid dynamics [3,4,236,237,239,240,338]:

Guideline 1. *A validation experiment should be jointly designed by experimentalists, model developers, code*

developers, and code users working closely together throughout the program, from inception to documentation, with complete candor about the strengths and weaknesses of each approach.

No withholding of limitations or deficiencies is permitted, and failure or success of any part of the effort must be shared by all. Without this level of cooperation, openness, and commitment in a team environment, the process is likely to fail or fall short of its potential.

Although Guideline 1 may sound easy to do, it is extraordinarily difficult to accomplish in practice. Some reasons for the difficulty have been discussed in Section 1. We give just two examples of why this is difficult to accomplish in reality between organizations and within an organization. First, suppose that the CFD team is in one organization and the wind-tunnel facility is in a completely separate organization, e.g., the wind-tunnel facility is contracted by the organization of the CFD team to conduct the experiments. The wind-tunnel facility will be extremely reluctant to expose its weaknesses, limitations, or deficiencies, especially if the facility was, or will be, in competition with other facilities to win contracts. In our experience, we have learned that validation experiments require a much greater depth of probing into the limitations of a facility and the limitations of a CFD capability than do any other types of experiments.

Second, suppose that the CFD team and the wind-tunnel team are sister organizations in the same corporation. Here also, both teams will be very cautious to discuss weaknesses in their CFD or experimental-facility capability. Because of the background and technical training of theoretical and experimental personnel, significant cultural hurdles must be overcome in dealing with one another. One of the most common detriments to achieving close teamwork and openness between computationalists and experimentalists is competition between the CFD team and the experimental team for funding or recognition. If it is advantageous for one of the teams to diminish the image or reputation of the other team in the "computers versus wind tunnel" mentality, there can be little chance for a true validation experiment. Management must make it clear, in word and deed, to both the CFD team and the experimental team that there is no success or failure of either side; there is only success or failure of the joint endeavor.

Guideline 2. *A validation experiment should be designed to capture the essential physics of interest, including all relevant physical modeling data and initial and boundary conditions required by the code.*

By essential physics of interest we mean spatial dimensionality, temporal nature, geometric complexity, and physical flow processes. For example, is one interested in conducting a two-dimensional experiment and computation, or is a full three-dimensional experiment and computation required? In this context, we note that no physical experiment can be truly planar two-dimensional; axisymmetric or spherical two-dimensional experiments are closely attainable. Experimentalists must understand the modeling assumptions embodied in the code so that the experiment can match, if possible, the assumptions and requirements. If the parameters that are initially requested for the calculation cannot be satisfied in the proposed experimental facility, it may be feasible to alter the code inputs and still satisfy the primary validation requirements. Or, it may be necessary to look elsewhere for a facility. For example, can the CFD-requested boundary-layer state on a model be ensured? Is the type and quantity of instrumentation appropriate to provide the required data in sufficient quantity and at the required accuracy and spatial resolution? Conversely, computationalists must understand the limitations of the physical experiment and facility, ensure that all the relevant physics are included, define physically realizable boundary conditions, and try to understand the formidable difficulties in attaining these conditions. As noted above, the level of detailed understanding that is required can be achieved only if the validation experiment is planned and conducted as part of a team effort.

The most common reason that published experimental data can be of only limited use in the validation of CFD codes is that insufficient detail is given in the documentation of the experiment concerning all of the needed physical modeling parameters, initial conditions, and boundary conditions. Published data in corporate reports or university reports occasionally contain sufficient characterization of the experiment, but rarely is sufficient detail contained in journal articles and conference papers. Following are a few examples of the level of detail that is needed for physical modeling parameters and boundary conditions on a vehicle in a wind tunnel:

- Accurate measurements (as opposed to nominal data) of the actual model dimensions (e.g., straightness and out-of-round).
- Surface roughness condition, including imperfections or mismatches in body components.
- Location of boundary layer transition for all free-stream conditions, angles of attack, and control surface deflections.
- Free-stream measurement of turbulence quantities.
- Accurate location of all instrumentation, as opposed to requested location stated in the fabrication drawings.
- For a subsonic free-stream, location of where the free-stream conditions were measured in the wind tunnel.

- For a subsonic free-stream, wall pressure measurements near the end of the computational domain, but inside the test section.
- Detailed dimensions of all model and instrumentation-mounting hardware.

For aircraft configurations, wings, and deformable bodies, an additional important detail is the measurement of the actual deformed geometry under the load, or an accurate calculation of the deformed structure. In long-duration hypersonic wind tunnels, the deformation of the vehicle due to aerodynamic heating should also be estimated or measured, and then reported. In addition, for hypersonic tunnels the model surface temperature should be measured at several locations during the run-time of the wind tunnel.

In wind tunnels, the highest priority for boundary conditions is probably calibration of the free-stream flow. This typically means spatially averaged quantities over the test-section volume of the tunnel, such as the free-stream Mach number, total pressure and static pressure, and total temperature and static temperature. For turbulent flow and boundary layer transition experiments, the calibration should also include free-stream turbulence intensity, scale, and frequencies. Some facility managers may be reluctant to share such detailed flow-quality data with users (and competitors). However, for experimental data that will be compared with CFD simulations that use sophisticated turbulence models or transition models, these data are critical.

For supersonic wind tunnels, experimental measurement of the spatial nonuniformity of the flow in the test section could be used to set the flow properties as location-dependent boundary conditions just upstream of the bow shock wave of the vehicle. To our knowledge, such an approach, although conceptually feasible, has not yet been demonstrated. Some might consider this approach excessive at this stage of CFD code development for validation experiments in wind tunnels of high-flow quality. However, we believe this approach is necessary for validation experiments in high-enthalpy-flow facilities in which rapid expansions combine with finite-rate chemistry. In such facilities, the flow is typically highly nonuniform and poorly characterized, which makes it extremely difficult, if not impossible, to accurately compare experimental data to code predictions.

For subsonic wind tunnels, the question of boundary conditions becomes much more complex. For low-speed wind tunnels, even with low levels of blockage, one of the first issues that must be addressed by the CFD analyst is, "Should I model the flow in the entire test section of the tunnel, or assume an infinite-size tunnel?" This question could be restated as, "For the quantity that will be measured in the tunnel and compared with the CFD computation, what is the change in this quantity if I assume a finite-size tunnel versus an infinite-size tunnel?" Although the authors have not seen any detailed analyses addressing this question, from our limited experience we believe that the sensitivity to tunnel blockage will be significant even at low blockage. For transonic flow tunnels, the large sensitivity of test-section measurements to solid walls versus perforated walls is well known. For perforated-wall tunnels, the tunnel-wall boundary conditions are very poorly characterized. We believe that a significant computational and experimental research program is needed to improve the mathematical modeling of perforated-wall tunnels. Unless this is done, the ability to conduct high-quality validation experiments in perforated-wall tunnels will greatly suffer. Solid-wall tunnels provide a well-characterized boundary condition for the CFD analyst, even if wall interference and wave reflections occur in the tunnel. This point emphasizes one of the key differences mentioned above concerning the difference between a test and a validation experiment: the *code* is the customer, not the project group whose only interest is in zero-wall interference.

Guideline 3. *A validation experiment should strive to emphasize the inherent synergism between computational and experimental approaches.*

By a "synergism," we mean an activity that is conducted by one approach, whether CFD or experiment, but which generates improvements in the capability, understanding, or accuracy of the other approach. And in this exchange, both computational and experimental approaches benefit. Some who are discovering the benefits of validation experiments claim that this is the primary value of validation experiments. Discovering the strong positive reinforcement of computationalists and experimentalists working closely together can be surprising, but validation experiments contribute much more than this. We give two examples of how this synergism can be exemplified.

First, the strength of one approach can be used to offset a weakness of the other approach. Consider the example of perfect-gas, laminar flow in a supersonic wind tunnel. Assume that a wind-tunnel model is designed so that it can be easily reconfigured from simple to complex geometries. For the simple geometry at a low angle of attack, one should be able to compute solutions with very high confidence, exclusive of the separated flow in the base region. This may require independent CFD codes and analyses, but it is certainly possible with present CFD technology. High-accuracy solutions can then be compared with wind-tunnel measurements to detect a wide variety of shortcomings and weaknesses in the facility, as well as errors in the instrumentation and data recording system. In our research we have demonstrated that the CFD computa-

tions can improve the understanding and characterization of our wind tunnel facility. The high accuracy solution can also be used for in situ calibration of the free-stream flow in the test section. When the complex vehicle geometry is tested, one may have strongly three-dimensional flows, separated flows, and shock wave/boundary-layer separation. The experimental measurements would then be more accurate than the CFD simulation, and the complex geometry case would then be viewed as a validation experiment to test the code.

Second, one can use CFD simulations in the planning stages of a validation experiment to dramatically improve the design, instrumentation, and execution of the experiment. For example, one can compute shock-wave locations and their impingement on a surface, separated flow and reattachment locations, regions of high heat flux, and vortical flows near a surface. Such computations allow the experimentalist to improve the design of the experiment and especially the type and the location of instrumentation. This strategy can also be taken a step further by optimizing the design of the experiment to most directly stress the code, i.e., design the experiment to "break the code." Optimizing the experimental design can be done by optimizing the physical modeling parameters, such as the Reynolds number and the Mach number; by modifying the boundary conditions, such as model geometry and wall-surface conditions; and also by changing the initial conditions on an initial-value problem. We should note that sometimes the code developers do not find the strategy of breaking their code too appealing.

Guideline 4. *Although the experimental design should be developed cooperatively, independence must be maintained in obtaining both the computational and experimental results.*

The reason for this recommendation, of course, is that it is so common for CFD codes to be calibrated to the experimental measurements that many people do not recognize when they are calibrating versus validating [12]. For example, computationalists have been known to say, "Why should I do any more grid refinement when the code (with its present experimentally determined quantities) agrees with the experimental data?" The discussion pertaining to the cancellation of errors and uncertainties was included in Section 4.1.

It is difficult to accomplish the close cooperation of the computationalists and the experimentalists and, at the same time, retain the independence of each group's results. However, this challenge can be met by careful attention to procedural details, as discussed next. Recall, first, that as we have stressed in preceding examples, a close working relationship is needed in the design and execution of the validation experiment. However, when the experimental measurements are reduced and analyzed, the CFD team should not be given the results initially. The CFD team should be given the complete details of the physical modeling parameters and the initial and boundary conditions of the experiment, *exactly* as it was conducted. That is, everything that is needed for the computationalists to compute solutions must be provided—but no more. The computationalists must quantify the errors and uncertainties in the CFD solution and then present the results for comparison with the experimental data. Then the validation test between the computational results and the experimental data is made. This test is performed jointly by the computationalists and the experimentalists.

After the comparison is analyzed and discussed, there can be various outcomes. If the error and uncertainty bounds on the computational side are very narrow, a large amount of data for estimating the mean values of the measurements is available, and the agreement is uniformly good, then one can crisply conclude that the computational results are validated. (Technically one cannot validate the code in a general sense: one only can claim "not invalidated.") In our experience, such crisp conclusions are rare. The more typical outcome is that there will be agreement on some measured quantities and there will be disagreement on other quantities. Or, one side or the other will say, "I need to go back and check a few things." After further checks, sometimes the agreement will improve; sometimes it will not. This discussion and iterative process is very beneficial to both sides of the team. We have found that the discussions will always lead to a deeper understanding of both the computational results and the experimental data.

As a final procedural comment, we recommend that management *not* be involved in the initial comparisons and discussions. The discussions should just involve computationalists and experimental staff. Nothing will poison the teamwork more quickly than one side of the team telling management, "Look how far off they were."

This guideline stresses the independence of the computational and experimental validation activities for an engineering environment, that is, validation of a CFD code that is intended for use in an engineering analysis environment, not a research code or a mathematical-model-building environment. For example, a code being developed by university researchers need not, in our view, attain the level of experimental and computational independence recommended here. Another example is the situation where experiments are being conducted for the purpose of building mathematical models of physical processes, e.g., turbulence models. As was pointed out earlier in this subsection, experiments to better understand physical processes are not true validation experiments, but are model-building experiments. Model-building experiments require very

close cooperation and communication between the model builder and the experimentalist; sometimes they are the same person.

Guideline 5. *A hierarchy of experimental measurements of increasing computational difficulty and specificity should be made, for example, from globally integrated quantities to local measurements.*

As one moves from global to locally measured quantities, the challenge to the computationalists and the experimentalists increases significantly. In wind-tunnel experimentation for a flight vehicle, the following hierarchy is suggested:

- Flight vehicle forces and moments.
- Control-surface forces and moments.
- Surface-pressure distributions.
- Surface heat flux, shear stress, or both.
- Flow-field distributions of pressure, temperature, and velocity components.
- Flow-field distributions of Reynolds stresses.

The ordering of difficulty in the above hierarchy indicates that each lower level, i.e., higher level of detail, is either the spatial or temporal derivative of the level above it, or it is a subset of the level above it. In other words, the integration or selection of a larger set is a powerful mathematical smoothing process. Thus, there is a relationship between this hierarchy and the levels of credibility in validation activities. That is, in the ''process of determining the degree to which a model is an accurate representation of the real world,'' one must specify what physical quantities, or system response measures, have been validated. For example, validation of body normal force *does not* imply that surface heat flux to the vehicle has been validated to the same degree of accuracy. There are two separate reasons for this statement. First, the physical modeling fidelity that is required to predict these two quantities is remarkably different. For example, body normal force on many geometries can be computed fairly accurately with inviscid flow, whereas the difficulty of predicting surface heat flux in a turbulent boundary layer is well known. Second, the computational grid size that is required to predict these two quantities is strikingly different. For example, consider a steady, compressible, laminar, attached flow over a delta wing at a low angle of attack. To achieve the same level of computational accuracy for body normal force as compared to surface heat flux, one would need approximately a factor of 100–1000 times the number of grid points for the heat-flux computation as compared to the normal-force computation.

The predictive difficulty for a code that is illustrated by the above hierarchy is also very similar to the difficulty in experimentally measuring each of these quantities. The experimental uncertainty increases as one proceeds down this list, probably at a factor of two for each level of the hierarchy. With the recent development of experimental techniques such as particle imaging velocimetry (PIV) and planar-laser-induced florescence (PLIF), we believe there is a significant increase in the quantity of available flow-field velocity measurements. The quantity of data, for example, over a large volume surrounding a body, permits much more stringent tests for CFD model validation than simply measurements at individual point locations.

The last recommendation concerning the hierarchy of measurements is that in a validation experiment one should, if possible, make measurements at multiple levels of the hierarchy. That is, do not design the experiment with the philosophy that only one detailed level of measurements will be made. The more complicated the flow field or the physical processes taking place in the flow field, the more important this recommendation becomes. For example, on a complex turbulent, reacting flow, do not just measure surface heat flux over a portion of the body. Also measure flow-field temperatures and surface pressures over a different portion of the body. Flow-field visualization and surface-flow visualization can also provide valuable additional pieces of information. With sophisticated postprocessing capability, the CFD solution can be used to simulate the experimental flow-field visualization and surface-flow visualization. For example, the computed flow-field solution can be used to compute a Schlieren or interferometer photograph that can then be compared with the experimental photograph.

Guideline 6. *The experimental design should be constructed to analyze and estimate the components of random (precision) and bias (systematic) experimental errors.*

The standard technique for estimating experimental uncertainty in wind-tunnel data has been developed over the last ten years by members of the AGARD Fluid Dynamics Panel [8]. The standard procedure, although not widely used in wind tunnel facilities, is well documented in a recent AIAA standards document [13] and also in the text of Coleman and Steele [75]. We believe it is the minimum level of effort required for uncertainty estimation in validation experiments. The standard technique propagates components of random and bias uncertainty through the entire data-flow process. The technique estimates these components and their interactions at each level of the process, from the sensor level to the experimental-result level. As with all techniques for estimating experimental uncertainty, the ability to estimate random uncertainty is much better than the ability to estimate bias uncertainty.

During the last 15 years a very different approach from the standard wind-tunnel procedure has been developed for estimating experimental uncertainty [3,236,237,239]. Instead of propagating individual uncertainty components through the data flow process we have taken an approach referred to in the AIAA standards document as an "end-to-end" approach. This approach compares multiple experimental measurements for the same experimental quantity and then statistically computes the uncertainty. This approach follows methods common in statistical model analysis. The traditional approach could be viewed as an a priori approach, whereas this is an a posteriori approach. Just as in comparing a priori and a posteriori error estimation in CFD, we believe this procedure provides not only a better estimate of random and bias errors but also a way to quantify important component contributions that cannot be estimated in the traditional approach. We discuss this new procedure in detail in the next subsection.

As a final comment on the estimation of experimental uncertainty, we recommend that the same validation experiment be conducted, if possible, in different facilities. For example, in a wind-tunnel experiment, the same physical model should be used and the experiment must be conducted at the same nominal free-stream conditions. Satisfactory agreement of results from different facilities lends significant confidence that there are no inadequately understood facility-related bias errors in the data, e.g., condensation effects, wave focusing, and excessive flow angularity. This procedure, especially for simple model geometries, would also serve to uncover inaccuracies and inconsistencies in the flow calibration data for each facility that is used. If the wind-tunnel model was susceptible to structural deflection due to aerodynamic loading, then each wind-tunnel entry should also be conducted at the same free-stream dynamic pressure. Consistent with the independence of computationalists and experimentalists stressed in Guideline 6, we recommend that there also be a level of independence between the personnel and the experimental measurements from each facility. When the same model is used in different facilities, surprising results are always discovered—usually to the dismay of the facility owner.

4.4. Statistical estimation of experimental error

A nontraditional approach for estimating random and bias experimental errors has been developed by several researchers at Sandia National Laboratories [3,236,237,239,245]. Although the approach is new to wind-tunnel testing, it is based upon accepted statistical uncertainty-estimation techniques [52,216]. This approach employs statistical methods to compute both the random and correlated bias uncertainties in the final experimental result. Symmetry arguments are applied to the flow in the test section and to the symmetry of the model, and then carefully selected comparison runs are conducted in order to gather the needed statistics. The approach is based on the following facts: (1) that a uniform free-stream flow field has an infinite number of planes of symmetry, and (2) that a perfectly constructed wind-tunnel model commonly has one plane of symmetry and some models have a larger number of planes of symmetry. For example, a typical aircraft has one plane of symmetry, and a four-finned missile has four planes of symmetry. Using these symmetry features and then comparing certain experimental measurements in the real wind tunnel and on the real model, one can statistically compute these correlated bias-error contributions to the total uncertainty.

Even though this nontraditional approach improves the estimate of the total random uncertainty compared to the traditional approach, we believe the most important contribution of this nontraditional approach is that it allows the estimate of correlated bias errors. The damaging effect of bias errors was sharply emphasized by Youden [348] in his classic paper. Youden pointed out that systematic errors commonly exceed the estimate of random errors, yet the magnitude of systematic errors is normally unknown. As an example, he listed 15 experimental measurements of the Astronomical Unit (AU) over the period from 1895–1961. Each experimentalist estimated the total uncertainty in his or her measurement, and in every case the next measurement made of the AU was *outside* the experimental uncertainty of the predecessor. Youden stated, "If we have accurate knowledge of the magnitude of the systematic errors in the components of the equipment, much of the discrepancy among results from different investigators would be accounted for." According to Youden, the most effective method for estimating systematic errors is to conduct experiments by multiple investigators, with different equipment, and with different techniques. This method, of course, is quite expensive and time consuming.

We have demonstrated this approach on four different sets of experimental data: three of the data sets were for surface-pressure measurements and one was for body forces and moments. The method has been applied to three hypersonic wind tunnels: Tunnels A and B of the von Karman Gas Dynamics Facility at the US Air Force Arnold Engineering Development Center, Tullahoma, Tennessee, and the Hypersonic Wind Tunnel Facility at Sandia National Laboratories, Albuquerque, New Mexico. The method showed that, even in these high-quality flow-field facilities, the largest contributor to experimental uncertainty was due to nonuniformity of the flow field. It was shown that the flow-field nonuniformity could be up to three standard deviations higher than the random (total

instrumentation) uncertainty. In terms of the percent of total (random and bias) estimated uncertainty, the flow-field nonuniformity can be up to 90% of the total, whereas the random uncertainty is 10%.

The method relies on careful construction and execution of the wind-tunnel run matrix so that combinations of runs yield information on both random and bias errors. For measurements of body force and moments, we refer to the random uncertainty as the uncertainty caused by all of the following components and their interactions: strain-gage hysteresis, nonlinearity, thermal sensitivity shift, and thermal zero shift; the analog data-reduction system; the data recording system; model pitch, roll, and yaw alignment; imperfections in model geometry; run-to-run variations in setting free-stream conditions in the test section; and base-pressure transducers and instrumentation for correcting for base drag. That is, the random uncertainty, when measuring body forces and moments, combines all uncertainty components in the entire experiment except those due to nonuniformity of the flow field in the test section. In [13], the statistical estimate of uncertainty is referred to as an end-to-end estimate of random uncertainty in the experimental result. To calculate the random uncertainty, one compares all possible combinations of body force and moment measurements that are made for the same physical location in the test section. This type of run-to-run comparison is referred to as a repeat-run comparison. Immediately repeating a particular case yields statistical information on short-term facility repeatability. Repeating runs in varying order, on different days, after the model has been disassembled and reassembled, and in separate facility entries, can uncover long-term facility repeatability. These long-term errors are related to facility operations, specific personnel, time of day, model assembly and installation repeatability, etc. Repeat runs require careful introspection in their selection and sequence and are critical to an assessment of statistical precision of the data. Repeat runs are not afterthoughts in this approach; they are essential elements in the method and must be incorporated into the experimental plan.

For an experiment measuring surface pressures on a body, we refer to the random uncertainty as that caused by all of the following random errors and their interactions with each other: pressure-sensor hysteresis, nonlinearity, sensitivity drift, and zero shift; variation in reference pressure; variation in the analog amplifier system; variation in the data digitizing and recording system; variation in model pitch, roll and yaw alignment; variations in the free-stream Mach number and Reynolds number within a run; variations in the free-stream Mach number and Reynolds number from run to run. The random uncertainty combines all experimental uncertainty in the entire experiment, except that due to nonuniformity of the flow field in the test section and that due to imperfections in model geometry. To calculate the random uncertainty, one compares pressure measurements for the same pressure port from different runs with the model at the same physical location and orientation in the test section. For the same angle of attack, roll angle, flap-deflection angle, and axial location, each pair of port measurements compared will have the same location in the vehicle-induced flow field. When differences in pressure-port measurements are made in this way, the uncertainty due to flow-field nonuniformity and to imperfections in model geometry cancels out.

The uncertainty due to nonuniformity of the flow field in the test section is the uncertainty in surface-pressure measurements caused by nonuniformity of free-stream flow in the test section and by bias errors in the alignment of the model in pitch, roll, and yaw. The uncertainty in an experimental measurement resulting from a combination of the uncertainty due to nonuniformity of the flow field in the test section and the random uncertainty is computed by comparing measurements made at different locations in the test section. For example, in an experiment measuring surface pressure, the combined flow-field nonuniformity and random uncertainty is calculated by comparing surface pressures for the same port on the body at the same relative location in the vehicle flow field, but at different locations in the test section. This procedure will not include any uncertainty due to model imperfections because by using the same ports for both comparisons, this uncertainty component cancels when the difference between the two port measurements is computed.

The uncertainty due to imperfections in model geometry can only be determined by measuring local surface quantities on a body of revolution, i.e., a body that has an infinite number of symmetry planes. For example, the uncertainty due to imperfections in model geometry in a surface-pressure experiment can be caused by the following: deviations in model geometry (measurable deviations of the physical model from the conceptual, or mathematical, description of the model) and imperfections in model/sensor installation (poorly fabricated or burred pressure orifice, or a pressure leak between the orifice and the transducer). The uncertainty due to imperfections in model geometry, along with the random uncertainty, is computed by comparing surface measurements for different transducers, with both transducers sensing at the same physical location in the test section and at the same relative location in the vehicle flow field. This requirement can only be met on bodies of revolution. This procedure will yield the combined model geometry and random uncertainty, but will not include any uncertainty due to flow-field nonuniformity.

The dominant contribution of nonuniform flow to experimental uncertainty has been suspected by

wind-tunnel experimentalists [13], but not until use of this approach has it been quantified. We strongly suspect that the largest contribution to measurement uncertainty in most, if not all, near-perfect-gas hypersonic wind tunnels is due to flow-field nonuniformity. Although this technique has not been applied to any other wind tunnel to our knowledge, we believe the dominance of flow-field-nonuniformity error will also occur in other facilities. We believe the nonuniform flow contribution will be even a higher percentage of the total experimental uncertainty in transonic flow wind tunnels and shock tunnels. Wind tunnel and shock tunnel facilities are encouraged to use the present statistical method to determine if this is the case. The critical assessment of one's own facility, however, will be viewed as a risk by many facility managers and owners. Some will choose to avoid the risk. We strongly believe that critical assessment of experimental uncertainty is just as important as critical assessment of computational error and uncertainty. Assessment of computational error was discussed in Section 3 and computational uncertainty is discussed in the next section. In Section 4.7, it will be shown that the accuracy of the experimental uncertainty is critical because it sets the limit on the quantitative assessment of validation.

4.5. Uncertainty quantification in computations

As mentioned in Section 4.1, it is common when simulating validation experiments that one encounters physical parameters (e.g., in the partial differential equations [PDEs] or in the initial or boundary conditions) that are not precisely known or measured for an experiment. This situation is much more common at the system and subsystem tiers than at the benchmark and unit-problem tiers. Examples of such parameters are thermochemical transport properties, flow rates in complex systems, and inflow nonuniformities. Another common situation occurs during a series of experiments, e.g., weather-driven experiments and flight tests, when there are certain parameters that are poorly controlled or not controllable at all. For any parameter, we make the assumption that a value of that parameter is required to perform the computational simulation. The situations just mentioned are all characterized by stochastic parametric uncertainty. In the following discussion, we simply refer to "uncertainty," but we will always be referring to parametric uncertainty. This type of uncertainty requires nondeterministic simulations in CFD validation.

One standard approach is to estimate, by one means or another, a single value of such a parameter and compute a solution with that selected value. This might be a fully adequate way of dealing with this uncertainty, especially if experience suggests that the range of potential parameter values is very small and that the calculation is known not to be extremely sensitive to the parameter in question. The resulting calculation intrinsically is interpreted as "typical," "representative," or "best estimate" for that parameter.

The shortcomings with the standard approach referred to above begin to be noticeable when the range of variation of the parameter is large or when the calculation is known to be sensitive to the parameter values. If multiple required parameters in a computation of a validation experiment are uncertain, then it is entirely possible that the interaction of these parameters in the calculation may magnify the influence of their uncertainty on the final results of the calculation. In our opinion, this statement is especially important when performing calculations that are intended for direct quantitative comparison with validation experiments. We believe that the uncertainty of the parameters should be incorporated directly into the computational analysis.

The simplest strategy for incorporating uncertainty of this kind directly into the computation is performed in three steps. The first step, called *characterizing the source* of uncertainty, is based on the assumption that the uncertainty in the parameters of interest is characterized by probability distributions. Sometimes such distributions can be directly estimated if a large quantity of experimental data is available for the parameters. Sometimes such distributions must simply be assumed. At any rate, the first step involves specifying the probability distributions and understanding the credibility of these assumptions.

In the second step, *ensemble computing*, values from the input probability distributions specified in the previous step are selected using statistical sampling procedures, such as Monte Carlo or Latin Hypercube sampling methods (see, for example, Ref. [83,126]). These sampled values are then used in a set of computations. Because this latter statement is so important, we will restate it for emphasis. The assumed prior probability distributions for the uncertain parameters are used to generate a set of calculations. This set is sometimes called an *ensemble* of calculations, and so this approach is referred to as *ensemble computing*.

The key issue is that a single calculation is no longer sufficient; a set of calculations must be performed. Obviously, this need is disturbing—where once one might have performed a single calculation, now one must perform a potentially large number of calculations. We have not raised nor answered the question of whether sufficient computational resources are available to execute more than the one calculation. However, the constraints enforced by availability of computing may be formidable.

After the set of calculations has been generated, the third step, *uncertainty quantification of the output*, is performed. This step involves analysis of the set of calculations, typically using statistical inference, to

estimate a probability distribution for the output variable(s) of interest that results from the given input parameter distributions. In general, we cannot deduce the exact output distribution that results from the assumed form of the parameter distributions used to generate the computational input associated with those parameters. Instead, the common practice is to use statistical procedures to determine estimates of important parameters associated with that output distribution.

Such statistically determined estimates are useful for comparing computational results with experimental measurements. For example, the mean of the output calculations provides us with an estimate of the expected value of the output quantity of interest, given the uncertainty structure specified by the input distributions. This mean-value estimate is of primary importance when comparing computational results with the mean value of multiple experimental realizations. Another statistic of interest is the estimated variance of the output distribution, which can be interpreted as a measure of computational output uncertainty, or scatter, given the input uncertainty.

For readers unfamiliar with this methodology, we stress that it is not true that the mean of the output given the input uncertainty can be determined by performing a calculation for a single set of inputs that is chosen to be the mean of each of the input distributions. Stated in another way, the mean value of the output cannot be computed by taking the mean value of all input parameters. Instead, we must perform the ensemble of calculations to develop a statistically rational estimator for the mean. The previous statement is also true for estimating other output statistics. Kleijnen [179] provides a broad summary of methodologies that go beyond Monte Carlo for assessing output distributions statistically.

In summary, the general methodology we are thus advocating for incorporating parameter uncertainty into CFD computations is to execute all three steps in the manner suggested above. Performing the three-step strategy will clearly be nontrivial for hard computational problems simply because of the computational burden imposed. There are also certain subtleties that we have failed to mention, such as whether complex structure in the resulting output probability distribution can actually be discovered using such a crude approach. We simply state that extracting an intricate output distribution structure will require either a large number of sample input values or considerably more sophisticated approaches for performing the methodology. The only fields we are aware of within CFD that pursue this methodology are the fields of underground transport of toxic waste materials or pollutants [35,153,156,196,322] and climatology [15,67,104,145,255,291].

One subtlety in performing uncertainty quantification was pointed out by Red-Horse and his colleagues [268]. As these researchers note, if we want to estimate, for example, the mean of the output quantity of interest using the methodology above, the estimated statistic that we determine is actually a *conditional mean*. In other words, the estimated mean of the output (given the uncertain input parameters that we calculate by the three-step strategy) assumes that the computational model is "correct," or valid, in a fundamental way. Note that this assumption is in addition to the operational assumption in step 1 that the input uncertainty distributions are "correct," or accurate enough, for the intended application of the computations. We thus have a coupling of the validation problem to the quantification of uncertainty in the computational model. To validate the computational model, we must characterize uncertainty quantitatively, using the methodology proposed above or something similar. However, for this characterization to be fully accurate requires a valid computational model. One way of looking at this coupling of the understanding of parameter sensitivity and model validation is through the use of Bayesian inference, which is briefly commented on further in this discussion.

It is also possible to remove the intrinsic foundation of a validated model in the estimation of output statistics by including fundamental model uncertainty, sometimes called model form uncertainty in the literature [103]. But can model uncertainty, which is far more general than the simple parameter uncertainty described previously, even be captured in a probabilistic framework? Nonprobabilistic approaches, referenced above with regard to epistemic uncertainty, are currently of interest for characterizing model uncertainty.

A closely related method for treating uncertainty is a sensitivity analysis [82,153,156,169,180,210,214,290]. A sensitivity analysis is typically composed of multiple simulations from a code to determine the effect of the variation of some component of the model, such as an input parameter or modeling assumptions, on output quantities of interest. Sometimes sensitivity analyses are referred to as "what-if" or perturbation analyses. Sensitivity analyses compute the rate of change of an output quantity with respect to an input quantity; all other input quantities remain fixed at a specified value. When the sensitivity of an output quantity is computed for a variety of input quantities, one can then rank the sensitivity of the output quantity with regard to the various input quantities. Although limited information is obtained with sensitivity analyses, sensitivity analyses are much less demanding computationally and in terms of required information as compared to uncertainty analyses.

We now return to our mention of Bayesian inference. In step 2 of the three-step strategy, the problem of propagating input uncertainty through a computational model to understand the resultant output as described

above, is sometimes referred to as the *forward uncertainty problem* [133]. There is an associated *inverse uncertainty problem*, or *backward problem*, which is conceptually and mathematically much more difficult. The backward problem asks whether we can reduce the output uncertainty by updating the statistical model using comparisons between computations and experiments. For example, might we be able to improve our original prior distributions that characterize the parameter uncertainty? This problem can be cast as a problem in Bayesian statistical inference [133]. (See Ref. [108] for an introduction to Bayesian inference.)

Part of the difficulty alluded to above is related to providing a better understanding of computational accuracy when it is known that we are in an under-resolved grid or time-step situation. This is an important research topic because of its practical consequences. Recent work attacks this problem and illustrates the formidable difficulties in two distinctly different areas: porous flow [132] and dynamical systems [68–70]. While the applications and specific technical attacks of researchers in these two areas are distinctly different, we find it fascinating that a common deep thread in their work is the treatment of insufficient information using statistical methods. We should stress, however, that these authors do not discuss one problem of interest to us—the problem of validation of under-resolved computational models. If reliable estimates of error due to under-resolved grids can be made, then we believe model validation can be conducted. The disadvantage, however, is that if large error bounds are estimated, then there can be a great deal of room for the computational results to agree, or disagree, with the experimental data. Stated differently, only *weak* validation conclusions can be drawn if either the computational or experimental error estimates are large. If no estimate of grid convergence error can be made for the computational solution, then we strongly believe *no* validation conclusion can be made. The same argument can be made if only one experimental measurement is made. However, it is traditional in engineering and science that relying on only one experimental measurement is more acceptable than not estimating grid convergence error, regardless of whether or not this position is actually defensible. The issue of how experimental uncertainty should be treated in validation is addressed in Section 4.7.

4.6. Hypothesis testing

As discussed in Section 4.1, validation quantification requires the determination and evaluation of a specified metric for measuring the consistency of a given computational model with respect to experimental measurements. One approach that has been traditionally taken in statistical analyses is hypothesis testing [193,197,226]. Hypothesis testing is a well-developed statistical method of choosing between two competing models of an experimental outcome by using probability theory to minimize the risk of an incorrect decision. In hypothesis testing the validation-quantification measure is formulated as a "decision problem" to determine whether or not the hypothesized model is consistent with the experimental data. This technique is regularly used in the operations research (OR) community for testing mutually exclusive models, i.e., the model is either true or false. For example, suppose the hypothesis is made that a coin is fair. That is, in the toss of the coin it is equally likely that "heads" will appear as often as "tails." The competing hypothesis is that the coin is unfair. Experimental data are then obtained by tossing the coin a given number of times, say N, and recording what percentage of the outcomes is heads and what percentage is tails. Hypothesis testing then allows one to statistically determine the confidence of a fair coin. The confidence in the determination will depend on N, that is, as N increases, the confidence in the conclusion increases.

Hypothesis testing has not been used to any significant degree in the validation quantification of computational physics. It seems there are two reasons for lack of interest in this approach. One reason is that validation of computational models of physical processes does not fit into the category of true or false hypotheses. For example, we would not expect to see it proclaimed, "Computer code *xyz* has been proven false!" Hypothesis testing would be more appropriate, for example, for testing whether Newtonian mechanics is true versus relativistic mechanics. In other words, model validation in the computational sciences is fundamentally an estimation process, *not* a true or false issue. For example, the appropriate questions in validation are: "What is the measure of agreement between the computational result and the experimental result?" "How much does the numerical error in the computational solution affect the measure of agreement?" and "How much does the experimental uncertainty affect the measure of agreement?"

A second reason for the lack of interest in hypothesis testing is that if this approach is used to prove a hypothesis true, given the available evidence, then the hypothesis, i.e., the model, can be used to replace the fiducial measure, i.e., the experiment. In the computational sciences, validation is properly understood to mean that the measure of agreement attained for one comparison case is an inference of validity for future cases, i.e., prediction. The accuracy of the prediction depends on many additional factors, such as the range of applicability of all of the submodels that compose the complete computational model, the change in coupling of the various physical processes from the validation case to the prediction case, the skill of the analyst in computing the prediction, and any additional

uncertainties that may enter into the prediction that were not present in the validation.

Even though the hypothesis-testing approach does not appear to be a constructive route forward for validation quantification, the approach has developed the concept of error types for incorrect conclusions drawn from hypothesis testing [193,197,226]. A type 1 error, also referred to as *model builder's risk*, is the error in rejecting the validity of a model when the model is actually valid. This can be caused by errors on both the computational side and the experimental side. On the computational side, for example, if a grid is not sufficiently converged and the computational result is in error, then an adverse comparison with experimental data is misleading. That is, a poor comparison leads one to conclude that a submodel, such as a turbulence or reacting flow model, needs to be improved or "recalibrated" when the source of the poor comparison is simply an under-resolved grid. On the experimental side, the model builder's risk is most commonly caused by a poor comparison of computational results and experimental data that is due to an unknown bias error in the experimental data. Examples of bias errors in wind-tunnel testing are the following: a bias error in the calibrated free-stream Mach number in the test section, a pressure reference value used in a differential pressure transducer drifts due to temperature of the transducer, and bias error due to flow-field nonuniformity and imperfections in model geometry (discussed above). We believe that unknown bias errors in experimental results are the most damaging in validation because if the experimental measurement is accepted, then it is concluded that the computational result is consistently in error; whereas in reality, the experiment is consistently in error. If the error is believed to be in the computation, then a great deal of effort will be expended trying to find the source of the error. Or worse, a computational submodel will be recalibrated using the biased experimental data. This results in transferring the experimental bias into the computational model and then biasing all future computations with the code.

The type 2 error, also referred to as *model user's risk*, is the error in accepting the validity of a model when the model is actually invalid. As with the type 1 error, this can be caused by errors on both the computational side and the experimental side. On the computational side, the logical reverse of the type 1 error described above can occur. That is, if a grid is not sufficiently converged and the computational result agrees well with the experiment, then the favorable comparison is also misleading. For example if a finer grid is used, one can find that the favorable agreement can disappear. This shows that the original favorable agreement has compensating, or canceling, errors in the comparison. We believe that compensating errors in complex simulations is a common phenomenon. Only the tenacious user of the code, or an uncommonly self-critical code developer, will dig deep enough to uncover the compensating errors. In a competitive or commercial code-development environment, such users or code developers as these can be very unpopular, and even muffled by co-workers and management. On the experimental side, the logical reverse of the type 1 error described above can occur. That is, if an unknown bias error exists in the experiment, and a favorable comparison between computational results and experimental data is obtained, the implication of code validity is incorrect. Similar to the type 2 error on the computational side, only the self-critical, and determined, experimentalist will continue to examine the experiment in an attempt to find any experimental bias errors.

Type 1 and type 2 errors are two edges of the same sword. In the OR literature, however, it is well known that model user's risk is potentially the more disastrous. The reason, of course, is that an apparently correct model (one that has experimental evidence that it is valid) is used for predictions and decision-making, when in fact it is incorrect. Type 2 errors produce a false sense of security. In addition to the potentially disastrous use of the model, we contend that the model user's risk is also the more likely to occur in practice than the model builder's risk. The reason is that with experimental evidence that the model is valid, there is little or no interest by analysts, experimentalists, managers, or funding sources to expend any more time or resources pursuing possible problems in either the computations or the experiments. Everyone is enthused by the agreement of results and "victory" is declared. Anyone who questions the results can risk loss of personal advancement and position within his or her organization.

4.7. Validation metrics

4.7.1. Recommended characteristics

Some of the authors referenced in Section 2.2 addressed validation quantification from a statistical viewpoint; that is, a probability distribution of a system response measure (due to input parameter uncertainty) is compared with a probability distribution due to experimental uncertainty. And as discussed in Section 4.5, the probabilistic response measure is commonly computed using Monte Carlo sampling so that a comparison with the experimental data can be made. Coleman and Stern [76] take a different approach. First, they include in their analysis an estimate of numerical solution error and its effect on the validation comparison. Second, they do not deal with the propagation of input probability distributions, but instead address the question of estimating the total uncertainty due to computation and experiment. And third, they define a metric for validation. If the difference between the

experimental measurement and the computational result is less than the validation metric, then the result is judged validated. Coleman and Stern's article added a new direction of thinking in validation quantification and validation metrics. However, we believe their approach is conceptually flawed in certain respects and we suggest a different approach [246].

We believe that a validation metric should have the following features:

(1) We agree with Coleman and Stern that the metric should incorporate an estimate of the numerical error in the computational simulation. However, we do not agree that this estimate should be grouped with the experimental uncertainty nor that it should be represented probabilistically. Numerical error represents a bias error in the solution, not a random distribution of a computational result around some mean value.
(2) The metric should not exclude any modeling assumptions or approximations used in the computation of the simulation result. That is, the computational result must reflect all uncertainties and errors incurred in the modeling and simulation process.
(3) We agree with Coleman and Stern that the metric should incorporate an estimate of the random errors in the experimental data, e.g., an estimate of the variance of an assumed Gaussian distribution. In addition, we believe the metric should also include an estimate of the correlated bias errors in the experimental data, as discussed in Section 4.4.
(4) The metric should depend on the number of experimental replications of a given measurement quantity. That is, the metric should reflect the level of confidence in the experimental mean that has been estimated, not just the variance or scatter in the data.
(5) The metric should be able to incorporate uncertainty in the computation that is due to both random uncertainty in experimental parameters and any uncertainty that is due to lack of experimental measurement of needed computational quantities. That is, the metric should use nondeterministic methods to propagate uncertainty through the computational model, as discussed in Section 4.5.

4.7.2. Validation metric example

To clarify our views on validation quantification and how we believe validation metrics should be constructed, we will discuss two closely related examples whose physics and mathematics are much simpler than fluid dynamics. We consider the example of a boundary-value problem described by a second-order, nonlinear ordinary differential equation (ODE). The one-dimensional domain of the boundary value problem and the boundary conditions are shown in Fig. 10. Let the general form of the ODE be given by

$$\frac{\mathrm{d}}{\mathrm{d}x}\left[p(x,y)\frac{\mathrm{d}y}{\mathrm{d}x}\right] + q(x,y)\frac{\mathrm{d}y}{\mathrm{d}x} + r(x,y) = 0, \tag{15}$$

where $p(x,y), q(x,y)$, and $r(x,y)$ are arbitrary functions of the independent variable, x, and the dependent variable, y. Let B_0 and B_L represent arbitrary boundary conditions at $x = 0$ and $x = L$, respectively. This ODE and its boundary conditions can represent many types of steady-state heat conduction in a one-dimensional solid. Examples are heat conduction in heterogeneous and anisotropic solids, temperature-dependent thermal conductivity, internal-heat generation, and convection and radiation heat loss along the solid. This ODE, of course, is intended to be analogous to multidimensional boundary-value problems in fluid dynamics.

The physical modeling parameters for this problem are incorporated into the functions p, q, and r, and also incorporated into B_0 and B_L. For example, in a heat conduction problem, if the thermal conductivity is a constant, then p is equal to the constant conductivity. Also for a heat conduction problem, if the boundaries are given as a specified temperature, then $y(0) =$ constant and $y(L) =$ constant are the boundary data for B_0 and B_L, respectively.

Let $Y(x_i)$ be the experimental measurements corresponding to the dependent variable, y, at $x = x_i$ for $i = 1, 2, 3, \ldots, I$. That is, a total of I locations are experimentally measured along the solid. Given this mathematical model and experimental data, we will consider two examples that emphasize different issues in validation quantification. For each example we suggest validation metrics that would be useful for various situations. For both examples we assume the following:

1. The numerical solution to the ODEs is without error. For an actual numerical solution, this practically means that the numerical solution error is carefully quantified, and it is shown that the relative error of the output quantities of interest is very small compared to experimental uncertainty.
2. There is no parametric uncertainty in the model. Stated differently, experimental measurements of p, q, r, B_0, and B_L have been made, they are assumed to be without measurement error, and they are used as deterministic input to the model.

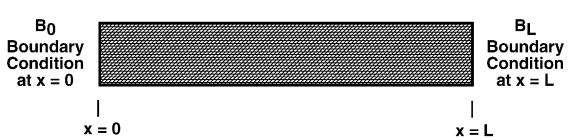

Fig. 10. Domain of boundary value problem.

3. The model and the experiment do not exhibit bifurcation or any chaotic phenomena.

Given assumption 1, we are avoiding the issue of how one conducts validation when numerical solution error is significant. As discussed in Section 4.1, we do not believe validation can be conducted when no estimate is available for the numerical solution error in the system responses that are compared with the experimental data. With assumption 2, we are restricting our examples to the issue of validating the fidelity of the mathematical model of the physical processes and avoiding two issues: first, the determination of the fidelity of parameters in the model, and second, statistical propagation of uncertainties when required code input parameters were not measured in the experiment. With techniques for propagation of uncertain parameters through the CFD model, discussed in Section 4.5, we are certain validation can still be conducted. However, validation becomes more complex because one must then deal with probability distributions for the computational inputs *and* responses. In the second example below, the issue of uncertainty in the experimental measurements is addressed.

4.7.3. Zero experimental measurement error

Since the experimental measurement error is zero, $y(x_i)$ and $Y(x_i)$ can be directly compared in different ways to generate different validation metrics. One useful validation metric based on comparison of each of the individual measurements and the computations at the measurement locations is

$$V = 1 - \frac{1}{I}\sum_{i=1}^{i=1} \tanh\left|\frac{y(x_i) - Y(x_i)}{Y(x_i)}\right|, \tag{16}$$

where V is the validation metric. This type of metric has the following advantages. First, it normalizes the difference between the computational results and the experimental data. Thus a relative error norm is computed. This normalization, however, is inappropriate when any of the $Y(x_i)$ are near zero. Second, the absolute value of the relative error only permits the difference between the computational results and the experimental data to accumulate, i.e., positive and negative differences cannot offset one another. Third, when the difference between the computational results and the experimental data is zero at all measurement locations, then the validation metric is unity, i.e., perfect agreement between the computational results and the experimental data. And fourth, when the summation of the relative error becomes large, the validation metric approaches zero.

Fig. 11 shows how the validation metric given in Eq. (16) varies as a function of constant values of the relative error at all spatial locations. As can be seen from Fig. 11, if the summation of the relative error is 100% of the experimental measurement, then the validation metric would yield a value of 0.239. If one chose a function different than $\tan h$ in Eq. (16), then the value of the metric for a constant error of 100% would, of course, be different. However, we do not believe the quantitative value of any metric is important in an absolute sense. By this we mean that the functional form of the metric is not absolute or unique. It only measures the agreement between the computational results and the experimental data in such a way that positive and negative errors cannot cancel. Our choice of the metric sets the value to be unity when perfect agreement is attained and an exponential decrease toward zero for very poor agreement of the computational results and the experimental data. Of prime importance is that the metric should be defined such that it addresses "the perspective of the intended uses of the model." Validation should not be viewed as binary issue, e.g., "Is the hypothesis true or false?" "Or is the computation within the scatter of the data, or not?"

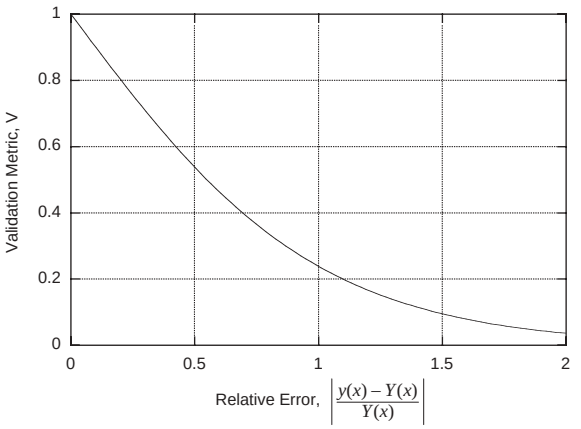

Fig. 11. Proposed validation metric as a function of relative error.

To clarify why a hypothesis-testing viewpoint is not as constructive, consider the following.

Let the hypothesis-testing validation metric be defined as

$$IF\left|\frac{y(x_i) - Y(x_i)}{Y(x_i)}\right| \begin{cases} \leqslant \varepsilon & \text{for all } i = 1, 2, 3, \ldots, I \Rightarrow \text{valid} \\ > \varepsilon & \text{for any } i = 1, 2, 3, \ldots, I \Rightarrow \text{invalid} \end{cases}, \tag{17}$$

where ε is a specified accuracy criteria. If a metric such as Eq. (17) is used, then the result is only "pass" or "fail." From an engineering viewpoint, validation is an estimation problem, which Eq. (17) does not address. From a scientific viewpoint, validation could be considered as a truth issue (see Ref. [251] and Appendix

C of Ref. [278]), in which case Eq. (17) might be more appropriate. We believe, however, that viewing validation as a philosophical issue will be of no value for engineering systems.

An additional argument against the type of metric given in Eq. (17) is that it merges the issue of validation accuracy and the question of whether the simulation is adequate for the intended uses of the model (commonly referred to as *qualification*). These are clearly separate issues. A useful validation metric should *only* measure the agreement between the computational results and the experimental data. Whether the measure of agreement is adequate for the intended uses must be viewed as a separate question, and indeed, the adequacy of the model for the intended uses, i.e., prediction, is the *more important* question. One of the conceptual difficulties is that there is coupling between the *choice* of the metric and the qualification acceptance requirement. Stated differently, there must be interaction between the choice (or specification) of the metric and the qualification acceptance requirement. Conversely, the qualification acceptance requirement should *not* influence the magnitude of a specified metric.

If a sufficient number of measurements is made along the solid, i.e., I is large in some sense, then one could accurately construct a continuous function to represent $Y(x)$ along the solid. For example, one could use a cubic spline to interpolate $[x_i, Y(x_i)]$ along the solid. Then one could construct an improved global-level metric analogous to Eq. (16):

$$V = 1 - \frac{1}{L}\int_0^L \tanh\left|\frac{y(x) - Y(x)}{Y(x)}\right| \mathrm{d}x. \tag{18}$$

If a sufficient number of experimental measurements are available to accurately construct the interpolation function, then this metric is more advantageous than Eq. (16) because it accounts for error in locations where experimental data are not available.

4.7.4. Random error in experimental measurements

Let there be N experimental measurements made at each of the x_i locations along the solid. Assume that the experimental random error is normally distributed, i.e., Gaussian, and assume that the measurement bias error is zero. The mean value of the N measurements at the position x_i is given by

$$\bar{Y}(x_i) = \frac{1}{N}\sum_{n=1}^{N} Y_n(x_i). \tag{19}$$

As N becomes large, the mean, $\bar{Y}(x_i)$, approaches the true value, $\tilde{Y}(x_i)$, of the Gaussian distributed measurements, that is,

$$\tilde{Y}(x_i) = \lim_{N\to\infty} \frac{1}{N}\sum_{n=1}^{N} Y_n(x_i). \tag{20}$$

Consider first the case when a large amount of data is available, i.e., N is large. Then the estimated mean and the true mean are effectively identical. $\tilde{Y}(x_i)$ is what should be compared with the computation, not its estimated uncertainty, *regardless* of the spread of the measurement distribution. Therefore, essentially the same validation metrics as given above in Example 1, where the data are perfect, can be used for this case. For example, the metric given in Eq. (18) would now be written as

$$V = 1 - \frac{1}{L}\int_0^L \tanh\left|\frac{y(x) - \tilde{Y}(x)}{\tilde{Y}(x)}\right| \mathrm{d}x. \tag{21}$$

$\tilde{Y}(x)$ would be the interpolated function constructed using $[x_i, \tilde{Y}(x_i)]$.

Now consider the case when a limited quantity of data is available. Assume that at each measurement position along the solid there are the same number of experimental measurements, N. The following suggested metric incorporates both the estimated variance of the data and the number of measurements at each station, N. Using the triangle inequality and taking expectations, one can extend the metric given in Eq. (21). One obtains

$$V = 1 - \frac{1}{L}\int_0^L \tanh, \left[\left|\frac{y(x) - \bar{Y}(x)}{\bar{Y}(x)}\right| + \int_{-\infty}^{\infty} \frac{s(x)}{\sqrt{N}}\left|\frac{z}{\bar{Y}(x)}\right| f(z)\,\mathrm{d}z\right]\mathrm{d}x, \tag{22}$$

where $s(x)$ is the sample standard deviation as a function of position along the solid, and $f(z)$ is the probability density function for a student's t-distribution with $N-1$ degrees of freedom. The integral inside the brackets is the expected absolute relative error of the experiment. The quantity $s(x)$ is approximated through the interpolated function constructed using $s(x_i)$, where

$$s^2(x_i) = \frac{1}{N-1}\sum_{n=1}^{N}[Y_n(x_i) - \bar{Y}_n(x_i)]^2. \tag{23}$$

It is obvious that no estimate of experimental uncertainty can be made if only one data point at each location, $N = 1$, is available.

The probability density function for the student's t-distribution is given by

$$f(z;\nu) = \frac{\Gamma\left((\nu+1)/2\right)}{\sqrt{\nu\pi}\,\Gamma(\nu/2)}\left[1 + \frac{z^2}{\nu}\right]^{-(\nu+1)/2}, \tag{24}$$

where ν is the number of degrees of freedom, $N-1$, and $\Gamma(\alpha)$ is the gamma function, given by

$$\Gamma(\alpha) = \int_0^{\infty} \xi^{\alpha-1}\,\mathrm{e}^{-\xi}\,\mathrm{d}\xi. \tag{25}$$

Fig. 12 shows the validation metric from Eq. (22) as a function of constant relative error for a coefficient of variation, $s(x)/Y(x) = 0.1$, and for various values of N.

The advantageous features of the validation metric given by Eq. (22) are the following. First, when a large

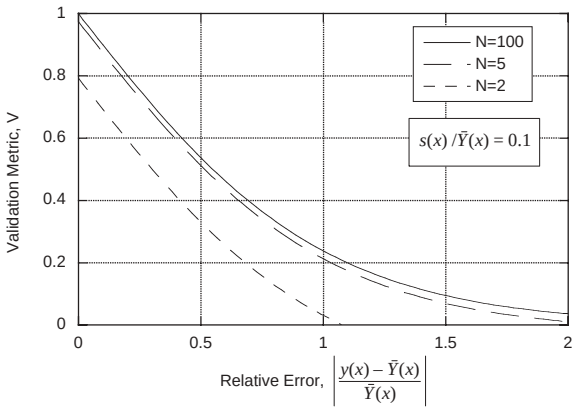

Fig. 12. Validation metric as a function of relative error and data quantity.

amount of data is available, i.e., N is large, the inner integral in Eq. (22) approaches zero. As a result, Eq. (22) approaches the metrics defined in Eqs. (18) and (21). Second, the student's t-distribution replaces the Gaussian distribution for Gaussian measurement errors when N is small. When N becomes large, the student's t-distribution approaches the Gaussian distribution. This makes the validation metric consistent with the assumed probability distribution for the random measurement error. Third, when N is small, the inner integral in Eq. (22) increases the magnitude of the integrand of the first integral. This results in decreasing the magnitude of the validation metric. That is, when small quantities of data are available, the validation metric should reflect the fact that confidence in the validation measure is decreased, and fourth, as the coefficient of variation in the data increases, for a given number of measurements N, the magnitude of the validation metric decreases. That is, as the spread of the data increases, the confidence in the validation measure is decreased.

5. Recommendations for future work and critical implementation issues

Rigorous and generalized procedures for verification and validation in CFD are in the early stages of development. While the foundational terminology, philosophy, and procedures required to formalize V&V are fairly well established, much work is required to develop the detailed computational and experimental procedures needed for efficient and quantifiable V&V. These procedures should be directed toward increasing confidence in the predictive capability of the computational model for a given level of effort and cost expended on V&V activities. In order to maximize the efficiency of these activities, widely applicable measures for quantification of V&V must be developed. Only with these quantitative measures can we begin to realize optimum benefits from the resources invested. Towards achieving increased efficiency, we offer several detailed recommendations.

Although a priori error estimation is useful for simplified problems, we believe it has very limited value for estimating the solution accuracy of complex CFD simulations. As an alternative, we recommend that increased mathematical research be pursued in a posteriori error estimation. It is our view that a posteriori error estimation is the only method that provides direct evidence of the performance of the numerical algorithm: for a particular system of nonlinear PDEs, for a particular set of initial and boundary conditions, and for a particular computer code. A posteriori error estimation based on multiple grid solutions is the most reliable type of error estimation, but error estimation based on comparing solutions on the same grid using methods with different orders of accuracy is computationally much more efficient.

Regarding the verification assessment techniques that stress calculation verification, we believe further development is needed in two areas. First, a wider variety of nonlinear analytical and semi-analytical solutions should be developed. For many decades applied mathematicians have concentrated on deriving analytical and semi-analytical solutions that are related to physical problems in fluids engineering. We recommend a shift in emphasis toward deriving more complex solutions of the nonlinear PDEs in CFD and de-emphasize the relationships of these solutions to particular fluids engineering problems. These complex solutions can then be used as verification benchmark solutions for nonlinearities related to physical phenomena such as shock waves, combustion, detonation, turbulence, two-phase flow, and chaotic solutions. Second, the method of manufactured solutions (MMS) should be further developed and applied for a wider variety of flow-modeling assumptions. For example, manufactured solutions should be investigated for supersonic flows with shock waves, turbulent flows, reacting flows, multiphase flows, and large eddy-simulation models. Although many code developers doubt the benefits of MMS, we have found that everyone who has used the technique is now a strong believer in its power. Two additional areas of MMS that require further research are (1) correct imposition of boundary conditions for mixed elliptic/parabolic/hyperbolic systems, and (2) determination of the types of errors that are *not* detectable by individual applications of MMS. Stated differently, the limits of MMS should be investigated more thoroughly so that any redundancy with traditional analytical-solution verification testing could be better understood.

The construction and use of a validation hierarchy for complex engineering systems is relatively new, but we believe this approach will be fundamentally useful in the future. In fact, we believe it is the *only* constructive approach for decomposing complex systems that can build demonstrable confidence in individual computational models. Consequently, we recommend that organizations that use computational modeling in the design, certification, or production of engineered systems should begin constructing and using the validation hierarchical structure. Admittedly, effective validation hierarchies—those that emphasize single physics phenomena at the lower levels and engineered system performance priorities at the higher levels—are difficult to construct. These hierarchies can only be constructed in an iterative fashion with input from a wide variety of professionals, from engineering analysts, design engineers to product marketing staff. In our limited use of the approach, we have found that a validation hierarchy quickly points out areas where little or no validation evidence exists. The ability of this approach to elucidate insufficiency or weakness in validation evidence may not be appealing to some advocates of computational simulation, but knowledge of such deficiencies is essential from an engineering perspective.

Validation experiments conducted at the higher levels of the hierarchy (subsystem and complete system levels) will invariably have poorly characterized experimental data for input to the computational simulation. As a result, validation will necessitate probabilistic treatment of uncertain parameters in the CFD submodels or in the initial conditions or boundary conditions for the PDEs. Propagation of these parameters or conditions through the CFD model will likely rely on probabilistic sampling methods like Monte Carlo or Latin Hypercube sampling. Importantly, major increases in computational resources will be required for the tens or hundreds of solutions needed for the sampling techniques. Additionally, improved training and understanding for probabilistic sampling techniques will be required for the CFD analysts involved in statistical validation activities. On this topic, we believe that the CFD community can learn a great deal from the probabilistic structural dynamics and risk assessment communities and can use computational tools developed by these communities.

The understanding of the unique characteristics of validation experiments, as opposed to traditional types of experiments, is slowly developing. Most of our guidelines presented for validation experiments will be difficult to implement and accept by experimental facilities. Implementation of certain guidelines will be difficult because some facility personnel will argue that the added costs of validation experiments will put their facility at a cost disadvantage compared to other facilities that do not incorporate the guidelines. Other personnel will argue that the guidelines will put their facility at risk because competitors will learn about their facility's weaknesses and shortcomings. These kinds of arguments reflect real-world business concerns, which we respect. Therefore, it is our belief that the only way the guidelines will be implemented by experimental facilities is if potential customers of these facilities require the use of the guidelines to win their validation experiment business.

The quantification of the level of agreement between computational results and experimental data, which we referred to as a validation metric, is an important topic for research. We have recommended beneficial characteristics for validation metrics and have suggested a metric that addresses some of these recommended characteristics. The suggested metric can be extended to two- and three-dimensional steady-state flows. Validation metrics are needed that weight important regions of the flow more heavily than less important regions, and metrics are needed for unsteady flows. If unsteady flows are periodic, then we believe that the metrics should be formulated not in the time domain, but in the frequency domain. As is well known from structural dynamics, the frequency domain eliminates the unimportant feature of phase shifts in the periodic response and concentrates on natural modes in the system.

We have emphasized that the definition and implementation of formal, high-quality V&V standards are likely to result in a significant improvement in CFD code confidence. Such standards would address the selection, use, and, most importantly, definition of assessment (pass/fail) criteria for V&V tests for CFD codes. We believe the most appropriate organizations for defining these standards are professional engineering societies that have standards-writing committees. The engineering societies could work cooperatively with national and international standards organizations such as the US National Institute of Standards and Technology (NIST) and the International Organization for Standardization (ISO), respectively. V&V standards documents and the construction of V&V databases would substantially help to formalize levels of CFD software capability and associated confidence, particularly for commercial CFD software. The road to developing V&V standards will be long and difficult, but we believe it is necessary as reliance on CFD software increases in the future.

Implementation of most of the approaches and procedures recommended here, for V&V experiments as well as computations, will be neither inexpensive nor easy. Furthermore, some of these approaches may even be technically or economically impractical in particular situations. With each included step, however, the quality of the code V&V processes will be improved. We firmly believe that V&V is a process, not a product. To achieve the level of maturity in CFD characterized by the US

National Research Council as "value exceeds expectation and most analyses are done without supporting experimental comparisons" [231] will require a much deeper understanding of mathematics, physics, computation, experiment, and their relationships than is reflected in current V&V practice.

Acknowledgements

The authors sincerely thank Frederick Blottner, Gary Froehlich, and Martin Pilch of Sandia National Laboratories, Patrick Roache, consultant, and Michael Hemsch of NASA/Langley Research Center for reviewing the manuscript and providing many helpful suggestions for improvement of the manuscript. We also thank Rhonda Reinert of Technically Write, Inc. for providing extensive editorial assistance during the writing of the manuscript. Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company for the US Department of Energy under contract No. DE-AC04-94AL85000.

References

[1] Abgrall R, Desideri JA. The European hypersonic database: a new CFD validation tool for the design of space vehicles. AIAA-93-3045, 1993.
[2] Aeschliman DP, Oberkampf WL. Experimental methodology for computational fluid dynamics code validation. SAND95-1189, Sandia National Laboratories, Albuquerque, NM, 1997.
[3] Aeschliman DP, Oberkampf WL. Experimental methodology for computational fluid dynamics code validation. AIAA J 1998;36(5):733–41.
[4] Aeschliman DP, Oberkampf WL, Blottner FG. A proposed methodology for CFD code verification, calibration, and validation, Paper 95-CH3482-7, 1995.
[5] AGARD. Experimental data base for computer program assessment. NATO Advisory Group for Aerospace Research & Development, AGARD-AR-138, 1979.
[6] AGARD. Experimental data base for computer program assessment: addendum. NATO Advisory Group for Aerospace Research & Development, AGARD-AR-138-ADDENDUM, 1984.
[7] AGARD. Aerodynamic data accuracy and quality: requirements and capabilities in wind tunnel testing. AGARD-CP-429, 1987.
[8] AGARD. Quality assessment for wind tunnel testing. NATO Advisory Group for Aerospace Research & Development (AGARD), AGARD-AR-304, 1994.
[9] AGARD. A selection of experimental test cases for the validation of CFD codes. NATO Advisory Group for Aerospace Research & Development, AGARD-AR-303-vol. II, 1994.
[10] AGARD. A selection of experimental test cases for the validation of CFD codes. NATO Advisory Group for Aerospace Research & Development, AGARD-AR-303-vol. I, 1994.
[11] Agonafer D, editor. Solutions to CFD benchmark problems in electronic packaging, HTD-vol. 255. New York: The American Society of Mechanical Engineers, 1993.
[12] AIAA. Guide for the verification and validation of computational fluid dynamics simulations. AIAA-G-077-1998, American Institute of Aeronautics and Astronautics, Reston, VA, 1998.
[13] AIAA. Assessment of experimental uncertainty with application to wind tunnel testing. S-071A-1999, American Institute of Aeronautics and Astronautics, Reston, VA, 1999.
[14] Ainsworth M, Oden JT. A posteriori error estimation in finite element analysis. New York: Wiley, 2000.
[15] Alapaty K, Raman S, Niyogi DS. Uncertainty in the specification of surface characteristics: a study of prediction errors in the boundary layer. Boundary-Layer Meteorol 1997;82(3):473–500.
[16] Almond RG. Graphical belief modeling, 1st ed. London, UK: Chapman & Hall, 1995.
[17] Alvin KF, Oberkampf WL, Rutherford BM, Diegert KV. Methodology for characterizing modeling and discretization uncertainties in computational simulation. SAND2000-0515, Sandia National Laboratories, Albuquerque, NM, 2000.
[18] Ambrosiano J, Peterson M-M. Research software development based on exploratory workflows: the exploratory process model (ExP). LA-UR-00-3697, Los Alamos National Laboratory, Los Alamos, NM, 2000.
[19] ANS. American Nuclear Society: Guidelines for the verification and validation of scientific and engineering computer programs for the nuclear industry, ANSI/ANS-10.4-1987, 1987.
[20] ASME. Council on codes, standards, Board of Performance Test Codes: Committee on Verification and Validation in Computational Solid Mechanics. American Society of Mechanical Engineers, www.usacm.org/vnvcsm/.
[21] Axelsson O. Iterative solution methods. Cambridge, UK: Cambridge Univ. Press, 1996.
[22] Ayyub BM. The nature of uncertainty in structural engineering. In: Ayyub BM, Gupta MM, editors. Uncertainty modelling and analysis: theory and applications. New York: Elsevier, 1994. p. 195–210.
[23] Baber R. The spine of software; designing provably correct software: theory and practice. New York: Wiley, 1987.
[24] Babuska I, Ihlenburg F, Strouboulis T, Gangaraj SK. A posteriori error estimation for finite element solutions of Helmholtz' equation—Part II: estimation of the pollution error. Int J Numer Methods Eng 1997;40:3883–900.
[25] Babuska I, Oh H-S. Pollution problem of the p- and h-p versions of the finite element method. Commun Appl Numer Methods 1987;3:553–61.
[26] Babuska I, Strouboulis T, Upadhyay CS, Gangaraj SK. A posteriori estimation and adaptive control of the pollution error in the h-version of the finite element method. Int J Numer Methods Eng 1995;38:4207–35.

[27] Bailey MP, Kemple WG. The scientific method of choosing model fidelity. 1992 Winter Simulation Conference Proceedings. Arlington, VA, 1992. p. 790–7.
[28] Balachandar S, Mittal R, Najjar FM. Properties of the mean recirculation region in the wakes of two-dimensional bluff bodies. J Fluid Mech 1997;351:167–99.
[29] Balci O, Nance RE. Formulated problem verification as an explicit requirement of model credibility. Simulation 452;1985:76–86.
[30] Balci O, Sargent RG. A bibliography on the credibility assessment and validation of simulation and mathematical models. Simuletter 1984;15(3):15–27.
[31] Banks J, Carson II. Discrete-event system simulation, 1st ed. Englewood Cliffs, NJ: Prentice-Hall, Inc., 1984.
[32] Barber TJ. Role of code validation and certification in the design environment. AIAA J 1998;36(5):752–8.
[33] Barkley D, Henderson RD. Three-dimensional floquet stability analysis of the wake of a circular cylinder. J Fluid Mech 1996;322:215–41.
[34] Barragy E, Carey GF. Stream function-vorticity driven cavity solution using p finite elements. Comput Fluids 1997;26(5):453–68.
[35] Beck MB. Water quality modeling: a review of the analysis of uncertainty. Water Resour Res 1987; 23(8):1393–442.
[36] Beizer B. Software testing techniques. New York: Van Nostrand Reinhold, 1990.
[37] Benek JA, Kraft EM, Lauer RF. Validation issues for engine—airframe integration. AIAA J 1998;36(5):759–64.
[38] Bertin JJ, Martellucci A, Neumann RD, Stetson KF. Developing a data base for the calibration and validation of hypersonic CFD codes—sharp cones. AIAA Paper No. 93-3044, 1993.
[39] Blackwell BF, Armaly BF, ediors. Computational aspects of heat transfer: benchmark problems, ASME HTD-vol. 258. New York: American Society of Mechanical Engineers, 1993.
[40] Blackwell BF, Pepper DW, editors. Benchmark problems for heat transfer codes, ASME HTD-vol. 222. New York: American Society of Mechanical Engineers, 1992.
[41] Blottner FG. Variable grid scheme applied to turbulent boundary layers. Comput Methods Appl Mech Eng 1974;4:179–94.
[42] Blottner FG. Investigation of some finite-difference techniques for solving the boundary layer equations. Comput Methods Appl Mech Eng 1975;6:1–30.
[43] Blottner FG. Introduction to computational techniques for boundary layers. SAND79-0893, Sandia National Laboratories, Albuquerque, NM, 1979.
[44] Blottner FG. Influence of boundary approximations and conditions on finite-difference solutions. J Comput Phys 1982;48(2):246–69.
[45] Blottner FG. Accurate Navier–Stokes results for the hypersonic flow over a spherical nosetip. J Spacecraft Rockets 1990;27(2):113–22.
[46] Blottner FJ, Lopez AR. Determination of solution accuracy of numerical schemes as part of code and calculation verification. SAND98-2222, Sandia National Laboratories, Albuquerque, NM, 1998.
[47] Boerstoel JW. Numerical accuracy assessment. AGARD-CP-437, 1988.
[48] Botella O, Peyret R. Computing singular solutions of the Navier–Stokes equations with the Chebyshev-collocation method. Int J Numer Methods Fluids 2001;36(2):125–63.
[49] Botta EFF, Dijkstra D, Veldman AEP. The numerical solution of the Navier–Stokes equations for laminar, incompressible flow past a parabolic cylinder. J Eng Math 1972;6(1):63–81.
[50] Bowen JP, Hinchey MG. 10-commandments of formal methods. Computer 1995;28(4):56–63.
[51] Bowen JP, Hinchey MG. Applications of formal methods. Englewood Cliffs, NJ: Prentice-Hall, 1995.
[52] Box GEP, Hunter WG, Hunter JS. Statistics for experimenters: an introduction to design, data analysis, and model building. New York: Wiley, 1978.
[53] Bradley RG. CFD validation philosophy. AGARD-CP-437, 1988.
[54] Bratley P, Fox BL, Schrage LE. A guide to simulation, 2nd ed. New York: Springer-Verlag, 1987.
[55] Bussoletti JE. CFD calibration and validation: the challenges of correlating computational model results with test data. AIAA-94-2542, 1994.
[56] Carnap R. Testability and meaning. Philosophy of science, vol. III, 1963.
[57] Carpenter MH, Casper JH. Accuracy of shock capturing in two spatial dimensions. AIAA J 1999; 37(9):1072–9.
[58] Casey M, Wintergerste T, editors. ERCOFTAC special interest group on quality and trust in industrial CFD: best practices guidelines. European Research Community on Flow, Turbulence and Combustion, 2000.
[59] Caughlin D. Verification, validation, and accreditation (VV&A) of models and simulations through reduced order metamodels. 1995 Winter Simulation Conference, Arlington, VA, 1995. p. 1404–12.
[60] Celik I, Hassan Y, Hughes D, Johnson R, Sommerfeld M, editors. Experimental and computational aspects of validation of multiphase flow CFD codes, FED-vol. 180. New York: The American Society of Mechanical Engineers, United Engineering Center, 1994.
[61] Celik I, Zhang WM. Calculation of numerical uncertainty using Richardson extrapolation: application to some simple turbulent flow calculations. J Fluids Eng 1995;117:439–45.
[62] Chaitin-Chatelin F, Fraysse V. Lectures on finite precision computations, Society for Industrial and Applied Mathematics. France: Toulouse, 1996.
[63] Chapman DR, Mark H, Pirtle MW. Computers vs. wind tunnels. Astronaut Aeronaut 1975;13(4):22–30.
[64] Checkland PB. Systems thinking, systems practice. New York: Wiley, 1981.
[65] Chiang TP, Sheu TWH. A numerical revisit of backward-facing step flow problem. Phys Fluids 1999;11(4): 862–74.
[66] Chiles J-P, Delfiner P. Geostatistics: modeling spatial uncertainty. New York: Wiley, 1999.
[67] Chlond A, Wolkau A. Large-eddy simulation of a nocturnal stratocumulus-topped marine atmospheric boundary layer: an uncertainty analysis. Boundary-Layer Meteorol 2000;95(1):31–55.
[68] Chorin AJ, Kast AP, Kupferman R. On the prediction of large-scale dynamics using unresolved computations.

LBNL-42283, Lawrence Berkeley National Laboratory, Berkeley, CA 1998
[69] Chorin AJ, Kast AP, Kupferman R. Optimal prediction of underresolved dynamics. Proc Natl Acad Sci 1998;95:4094–8.
[70] Chorin AJ, Kast AP, Kupferman R. Unresolved computation and optimal prediction. Commun Pure Appl Math 1999;52:1231–54.
[71] Churchman CW. The systems approach. New York: Dell, 1968.
[72] Clark Jr. EM, Grumberg O, Peled D. Model checking. Cambridge, MA: MIT Press, 1999.
[73] Cohen ML, Rolph JE, Steffey DL, editors. Statistics, testing, and defense acquisition: new approaches and methodological improvements. Washington, DC: National Academy Press, 1998.
[74] Cole JD, Cook LP, Schleiniger G. Analysis of a glancing shock moving past a wedge. AIAA-98-2688, 1998.
[75] Coleman HW, Steele Jr. WG. Experimentation and uncertainty analysis for engineers, 2nd ed. New York: Wiley, 1999.
[76] Coleman HW, Stern F. Uncertainties and CFD code validation. J Fluids Eng 1997;119:795–803.
[77] Conway RW. Some tactical problems in digital simulation. Manage Sci 1963;10(1):47–61.
[78] Cosner, RR. Issues in aerospace application of CFD analysis. AIAA Paper No. 94-0464, 1994.
[79] Cosner RR. CFD validation requirements for technology transition. AIAA Paper No. 95-2227, 1995.
[80] Cosner RR. The role of validation in the CFD process at McDonnell Douglas/St. Louis. AIAA-96-2273, 1996.
[81] Cosner RR. Experimental data needs for risk management in CFD applications. AIAA Paper No. 98-2781, 1998.
[82] Cukier RI, Levine HB, Shuler KE. Nonlinear sensitivity analysis of multiparameter model systems. J Comput Phys 1978;26(1):1–42.
[83] Cullen AC, Frey HC. Probabilistic techniques in exposure assessment: a handbook for dealing with variability and uncertainty in models and inputs. New York: Plenum Press, 1999.
[84] Dahl O. Verifiable programming. Englewood Cliff, NJ: Prentice-Hall, 1992.
[85] Davis PA, Olague NE, Goodrich MT. Approaches for the validation of models used for performance assessment of high-level nuclear waste repositories. NUREG/CR-5537; SAND90-0575, Sandia National Laboratories, Albuquerque, NM, 1991.
[86] Davis PK. Generalizing concepts and methods of verification, validation, and accreditation (VV&A) for military simulations. RAND, R-4249-ACQ, Santa Monica, CA, 1992.
[87] Davis RT. Laminar incompressible flow past a semi-infinite flat plate. J Fluid Mech 1967;27(4):691–704.
[88] Davis RT. Numerical solution of the Navier–Stokes equations for symmetric laminar incompressible flow past a parabola. J Fluid Mech 1972;51(3):417–33.
[89] de Vahl Davis G. Natural convection of air in a square cavity: a benchmark numerical solution. Int J Numer Methods Fluids 1983;3:249–64.
[90] Deiwert GS. Issues and approach to develop validated analysis tools for hypersonic flows: one perspective. NASA-TM-103937, NASA Ames Research Center, 1992.
[91] DeMillo RA, McCracken WM, Martin RJ, Passafiume JF. Software testing and evaluation. Menlo Park, CA: Benjamin/Cummings, 1987.
[92] Dennis SCR, Walsh JD. Numerical solutions for steady symmetric viscous flow past a parabolic cylinder in a uniform stream. J Fluid Mech 1971;50(4):801–14.
[93] Dery R, Landry M, Banville C. Revisiting the issue of model validation in OR: an epistemological view. Eur J Oper Res 1993;66:168–83.
[94] Desideri JA, Glowinski R, Periaux J, editors. Hypersonic flows for reentry problems, vol. I: Survey lectures and test cases for analysis. Berlin: Springer, 1991.
[95] Desideri JA, Glowinski R, Periaux J, editors. Hypersonic flows for reentry problems, vol. II: Test cases-experiments and computations. Berlin: Springer, 1991.
[96] Dietrich DE, Roache PJ, Marietta MG. Convergence studies with the Sandia Ocean Modeling system. Int J Numer Methods Fluids 1990;11(2):127–50.
[97] DiMascio A, Paciorri R, Favini B. Convergence of two numerical schemes for turbulent boundary layer computations. AIAA 98-3009, 1998.
[98] DoD. DoD Directive No. 5000.59: Modeling and simulation (M&S) management. Defense Modeling and Simulation Office, Office of the Director of Defense Research and Engineering, www.dmso.mil/docslib, 1994.
[99] DoD. DoD Instruction 5000.61: Modeling and simulation (M&S) verification, validation, and accreditation (VV&A). Defense Modeling and Simulation Office, Office of the Director of Defense Research and Engineering, www.dmso.mil/docslib, 1996.
[100] DoD. Verification, validation, and accreditation (VV&A) recommended practices guide. Defense Modeling and Simulation Office, Office of the Director of Defense Research and Engineering, www.dmso.mil/docslib, 1996.
[101] Dolling, DS. Problems in the validation of CFD codes through comparison with experiment. AGARD-CP-514, 1992.
[102] Dolling DS. High-speed turbulent separated flows: consistency of mathematical models and flow physics. AIAA J 1998;36(5):725–32.
[103] Draper D. Assessment and propagation of model uncertainty. J R Stat Soc B 1995;57(1):45–97.
[104] Du J, Mullen SL, Sanders F. Short-range ensemble forecasting of quantitative precipitation. Mon Weather Rev 1997;125(10):2427–59.
[105] Dubois D, Prade H. Possibility theory: an approach to computerized processing of uncertainty. New York: Plenum Press, 1988.
[106] Dwoyer D. The relation between computational fluid dynamics and experiment, AIAA Ground Testing Conf. 1992.
[107] Dyer M. The cleanroom approach to quality software development. New York: Wiley, 1992.
[108] Earman J. Bayes or bust? Cambridge, MA: The MIT Press, 1992.
[109] ERCOFTAC. Portal to fluid dynamics database resources. European Research Community on Flow,

Turbulence and Combustion, http://ercoftac.mech.surrey.ac.uk.
[110] Ethier CR, Steinman DA. Exact fully 3D Navier–Stokes solutions for benchmarking. Int J Numer Methods Fluids 1994;19:369–75.
[111] Evans LC. Partial differential equations. Providence, RI: American Mathematical Society, 1998.
[112] FAA. System design analysis. Federal Aviation Administration, Advisory Circular 25.1309-1A, Washington, DC, 1988.
[113] Fairley RE. Software engineering concepts. New York: McGraw-Hill, 1985.
[114] Ferson S, Ginzburg LR. Different methods are needed to propagate ignorance and variability. Reliability Eng System Saf 1996;54:133–44.
[115] Ferziger JH, Peric M. Computational methods for fluid dynamics. New York: Springer-Verlag, 1996.
[116] Ferziger JH, Peric M. Further discussion of numerical errors in CFD. Int J Numer Methods Fluids 1996;23:1263–74.
[117] Fletcher CAJ. Generating exact solutions of the two-dimensional burgers equations. Int J Numer Methods Fluids 1983;3(3):213–6.
[118] FLOWNET. Flow Library on the Web Network, FLOWNET Consortium-European Union, www-sop.inria.fr/sinus/flownet.
[119] Folland GB. Introduction to partial differential equations, 2nd ed. Princeton, NJ: Princeton University Press, 1996.
[120] Fortin A, Jardak M, Gervais JJ, Pierre R. Localization of HOPF bifurcations in fluid flow problems. Int J Numer Methods Fluids 1997;24(11):1185–210.
[121] Fossett CA, Harrison D, Weintrob H, Gass SI. An assessment procedure for simulation models: a case study. Oper Res 1991;39(5):710–23.
[122] Frank MV. Treatment of uncertainties in space nuclear risk assessment with examples from Cassini mission applications. Reliability Eng System Saf 1999;66:203–21.
[123] Freitas CJ, editor. The CFD triathlon: three laminar flow simulations by commercial CFD Codes, FED-vol. 160. New York: The American Society of Mechanical Engineers, United Engineering Center, 1993.
[124] Frey HC, Rhodes DS. Characterizing, simulating, and analyzing variability and uncertainty: an illustration of methods using an air toxics emissions example. Hum Ecol Risk Assess 1996;2(4):762–97.
[125] Gallopoulos E, Sameh A. CSE: content and product. IEEE Comput Sci Eng 1997;4(2):39–43.
[126] Gamerman D. Markov Chain Monte Carlo. London: Chapman & Hall, 1997.
[127] Gartling DK. A test problem for outflow boundary conditions-flow over a backward-facing step. Int J Numer Methods Fluids 1990;11:953–67.
[128] Gass SI. Model Accreditation: a rationale and process for determining a numerical rating. Eur J Oper Res 1993;66:250–8.
[129] Gerbeau JF, LeBris C, Bercovier M. Spurious velocities in the steady flow of an incompressible fluid subjected to external forces. Int J Numer Methods Fluids 1997;25(6):679–95.
[130] Gervais JJ, Lemelin D, Pierre R. Some experiments with stability analysis of discrete incompressible flow in the lid-driven cavity. Int J Numer Methods Fluids 1997;24(5):477–92.
[131] Ghia U, Ghia KN, Shin CT. High-Re solutions for incompressible flow using the Navier–Stokes equations and a multigrid method. J Comput Phys 1982;48(3):387–411.
[132] Glimm J, Hou S, Kim H, Sharp DH, Ye K. A probability model for errors in the numerical solutions of a partial differential equation. LAUR-99-5352, Los Alamos National Laboratory, Los Alamos, NM, 1999.
[133] Glimm J, Sharp DH. Stochastic methods for the prediction of complex multiscale phenomena. LAUR-97-3748, Los Alamos National Laboratory, Los Alamos, NM, 1997.
[134] Goldstein D, Hughes D, Johnson R, Lankford D, editors. Data for validation of CFD codes, FED-vol. 146. New York: The American Society of Mechanical Engineers, United Engineering Center, 1993.
[135] Gosman AD. Quality assurance for industrial CFD codes. AIAA 98-2637, 1998.
[136] Graves Jr. R. Software validation: the bridge from R&D to industrial application. AIAA-92-0587, 1992.
[137] Grenda JM, Schwer DA, Merkle CL. Use of analytical solutions in validating unsteady CFD Codes. AIAA-96-2164, 1996.
[138] Gresho PM, Gartling DK, Torczynski JR, Cliffe KA, Winters KH, Garratt JT, Spence A, Goodrich JW. Is the steady viscous incompressible two-dimensional flow over a backward-facing step at Re = 800 stable? Int J Numer Methods Fluids 1993;17:501–41.
[139] Grinstein FF. Open boundary conditions in the simulation of subsonic turbulent shear flows. J Comput Phys 1994;115:43–55.
[140] Guan J, Bell DA. Evidence theory and its applications. Amsterdam: North Holland, 1991.
[141] Guerrero JSP, Cotta RM. Integral transform solution for the lid-driven cavity flow problem in streamfunction-only formulation. Int J Numer Methods Fluids 1992;15(4):399–409.
[142] Guerrero JSP, Cotta RM. Benchmark integral transform results for flow over a backward-facing step. Comput Fluids 1996;25(5):527–40.
[143] Gustafson J. Computational verifiability and feasibility of the ASCI program. IEEE Comput Sci Eng 1998;5(1):36–45.
[144] Gustafson K. Capturing correct solutions in CFD, 515. Sixteenth International Conference on Numerical Methods in Fluid Dynamics. Arcachon, France, 1998. p. 171–6.
[145] Hamill TM, Wilks DS. A probabilistic forecast contest and the difficulty in assessing short-range forecast uncertainty. Weather Forecasting 1995;10(3):620–31.
[146] Hamilton MA. Model validation: an annotated bibliography. Commun Stat-Theory Methods 1991;20(7):2207–66.
[147] Hansen EB, Kelmanson MA. An integral equation justification of the boundary conditions of the driven-cavity problem. Comput Fluids 1994;23(1):225–40.

[148] Hanson KM. A framework for assessing uncertainties in simulation predictions. Physica D 1999;133:179–88.
[149] Hatton L. The T experiments: errors in scientific software. IEEE Comput Sci Eng 1997;4(2):27–38.
[150] Haworth DC, Tahry SHE, Huebler MS. A global approach to error estimation and physical diagnostics in multidimensional computational fluid dynamics. Int J Numer Methods Fluids 1993;17(1):75–97.
[151] Hayes WD, Probstein RF. Hypersonic flow theory: Inviscid flows, vol. 1. New York: Academic Press, 1966.
[152] Haynes TS, Reed HL, Saric WS. CFD validation issues in transition modeling. AIAA-96-2051, 1996.
[153] Helton JC. Uncertainty and sensitivity analysis techniques for use in performance assessment for radioactive waste disposal. Reliability Eng System Saf 1993; 42(2–3):327–67.
[154] Helton JC. Treatment of uncertainty in performance assessments for complex systems. Risk Anal 1994;14(4): 483–511.
[155] Helton JC. Uncertainty and sensitivity analysis in the presence of stochastic and subjective uncertainty. J Stat Comput Simul 1997;57:3–76.
[156] Helton JC. Uncertainty and sensitivity analysis in performance assessment for the waste isolation pilot plant. Comput Phys Commun 1999;117(1–2):156–80.
[157] Hills RG, Trucano TG. Statistical validation of engineering and scientific models with application to CTH. SAND2001-0312, Sandia National Laboratories, Albuquerque, NM, 2001.
[158] Hirsch C. Numerical computation of internal and external flows, vol. 1: Fundamentals of numerical discretization. New York, NY: Wiley, 1988.
[159] Hirsch C. Numerical computation of internal and external flows, vol. II: Computational methods for inviscid and viscous flows. New York: Wiley, 1990.
[160] Hodges JS, Dewar JA. Is it you or your model talking? A framework for model validation. RAND, R-4114-AF/A/OSD, Santa Monica, CA, 1992.
[161] Hoffman FO, Hammonds JS. Propagation of uncertainty in risk assessments: the need to distinguish between uncertainty due to lack of knowledge and uncertainty due to variability. Risk Anal 1994;14(5):707–12.
[162] Holden MS, Moselle JR. A database of aerothermal measurements in hypersonic flow for CFD validation. AIAA Paper No. 92-4023, 1992.
[163] Holden MS, Moselle JR, Sweet SJ, Martin SC. A database of aerothermal measurements in hypersonic flow for CFD validation. AIAA Paper No. 96-4597, 1996.
[164] Hora SC. Aleatory and epistemic uncertainty in probability elicitation with an example from hazardous waste management. Reliability Eng System Saf 1996; 54:217–23.
[165] Hortmann M, Peric M, Scheuerer G. Finite volume multigrid prediction of laminar natural convection: bench-mark solutions. Int J Numer Methods Fluids 1990;11(2):189–207.
[166] Hutton AG, Casey MV. Quality and trust in industrial CFD—a European perspective. AIAA2001-0656, 2001.
[167] IEEE. IEEE standard dictionary of electrical and electronics terms. ANSI/IEEE Std 100-1984, 1984.
[168] IEEE. IEEE standard glossary of software engineering terminology. IEEE, IEEE Std 610.12-1990, New York, 1991.
[169] Iman RL, Helton JC. An investigation of uncertainty and sensitivity analysis techniques for computer models. Risk Anal 1988;8(1):71–90.
[170] ISO. ISO 9000-3: Quality management and quality assurance standards—Part 3: guidelines for the application of ISO 9001 to the development, supply and maintenance of software. Geneva, Switzerland: International Standards Organization, 1991.
[171] Jameson A, Martinelli L. Mesh refinement and modeling errors in flow simulation. AIAA J 1998;36(5):676–86.
[172] Johnson DM. A review of fault management techniques used in safety-critical avionic systems. Prog Aerosp Sci 1996;32(5):415–31.
[173] Jones C. Applied software measurement, 2nd ed. New York: McGraw-Hill, 1997.
[174] Kammeyer ME. Wind tunnel facility calibrations and experimental uncertainty. AIAA 98-2715, 1998.
[175] Kaner C, Falk J, Nguyen HQ. Testing computer software, 2nd ed. New York: Wiley, 1999.
[176] Karamcheti K. Principles of ideal-fluid aerodynamics. New York: Wiley, 1973.
[177] Karniadakis GE, Triantafyllou GS. Three-dimensional dynamics and transition to turbulence in the wake of bluff objects. J Fluid Mech 1992;238:1–30.
[178] Ketelle RH, Lee RR, Bownds JM, Rizk TA. Model validation lessons learned: a case study at Oak Ridge National Laboratory. CONF-89085406/DE89 015900, Oak Ridge National Laboratory, Oak Ridge, TN, 1989.
[179] Kleijnen JPC. Statistical tools for simulation practitioners, 1st ed. New York: Marcel Dekker, Inc., 1987.
[180] Kleijnen JPC. Sensitivity analysis versus uncertainty analysis: when to use what? In: Grasman J, van Straten G, editors. Predictability and nonlinear modelling in natural sciences and economics. Boston: Kluwer Academic, 1994. p. 322–32.
[181] Kleijnen JPC. Statistical validation of simulation models. Eur J Oper Res 1995;21–34.
[182] Kleijnen JPC. Verification and validation of simulation models. Eur J Oper Res 1995;82:145–62.
[183] Kleindorfer GB, O'Neill L, Ganeshan R. Validation in simulation: various positions in the philosophy of science. Manage Sci 1998;44(8):1087–99.
[184] Klir GJ, Folger TA. Fuzzy sets, uncertainty, and information, 1st ed. Englewood Cliffs, NJ: Prentice Hall, 1988.
[185] Klir GJ, Wierman MJ. Uncertainty-based information: elements of generalized information theory. Heidelberg: Physica-Verlag, 1998.
[186] Knepell PL, Arangno DC. Simulation validation: a confidence assessment methodology, 1st ed. Washington, DC: IEEE Computer Society Press, 1993.
[187] Kohlas J, Monney P-A. A mathematical theory of hints—an approach to the Dempster–Shafer theory of evidence. Berlin: Springer-Verlag, 1995.
[188] Kurshan RP. Program verification. Notices Am Math Soc 2000;47(5):534–45.

[189] Landry M, Malouin J-L, Oral M. Model validation in operations research. Eur J Oper Res 1983;14:207–20.
[190] Landry M, Oral M. In search of a valid view of model validation for operations research. Eur J Oper Res 1993;66:161–7.
[191] Laney CB. Computational gas dynamics. New York: Cambridge University Press, 1998.
[192] Laskey KB. Model uncertainty: theory and practical implications. IEEE Trans Systems, Man and Cybernetics-Part A: Systems Hum 1996;26(3):340–8.
[193] Law AM, Kelton WD. Simulation modeling and analysis, 2nd ed. New York: McGraw-Hill, 1991.
[194] Lee LH, Poolla K. Statistical validation for uncertainty models. Lecture Notes in Control and Information Sciences 1994. p. 202.
[195] Lee LH, Poolla K. On statistical model validation, journal of dynamic systems. Meas Control 1996; 118:226–36.
[196] LeGore T. Predictive software validation methodology for use with experiments having limited replicability. In: Celik I, Freitas CJ, editors. Benchmark test cases for computational fluid dynamics, FED-vol. 93. New York: American Society of Mechanical Engineers, 1990. p. 21–7.
[197] Lehmann EL. Testing statistical hypotheses. New York: Wiley, 1986.
[198] Leijnse A, Hassanizadeh SM. Model definition and model validation. Adv Water Resour 1994;17:197–200.
[199] LeVeque RJ. Numerical methods for conservation laws. Basel, Switzerland: Birkhauser-Verlag, 1992.
[200] Lewis RO. Independent verification and validation, 1st ed. New York: Wiley, 1992.
[201] Lin SJ, Barson SL, Sindir MM. Development of evaluation criteria and a procedure for assessing predictive capability and code performance. Advanced Earth-to-Orbit Propulsion Technology Conference. Marshall Space Flight Center, Huntsville, AL, 1992.
[202] Mair HU. Benchmarks for submerged structure response to underwater explosions. Shock Vibration 1999; 6(4):169–81.
[203] Marciniak JJ, editor. Encyclopedia of software engineering. New York: Wiley, 1994.
[204] Martellucci A. The challenging process of validating CFD codes. AIAA-90-1402, 1990.
[205] Marvin JG. Accuracy requirements and benchmark experiments for CFD validation. AGARD-CP-437, 1988.
[206] Marvin JG. CFD Validation experiments for hypersonic flows. AIAA-92-4024, 1992.
[207] Marvin JG. Perspective on computational fluid dynamics validation. AIAA J 1995;33(10):1778–87.
[208] McDonald WW. Introduction to the submarine damage mechanisms project. Naval Surface Warfare Center, IHTR 1824, Indian Head, MD, 1995.
[209] McIntyre J, Sicilian J, Giezen J. A benchmark for inviscid incompressible flow in spinning containers. In: Celik I, Freitas CJ, editors. Benchmark test cases for computational fluid dynamics. New York: The American Society of Mechanical Engineers, 1990. p. 39–47.
[210] McKay MD. Aspects of modeling uncertainty and prediction. LANL-94-605, Los Alamos National Laboratory, Los Alamos, NM, 1994.
[211] Mehta UB. Computational requirements for hypersonic flight performance estimates. J Spacecraft Rockets 1990;27(2):103–12.
[212] Mehta UB. Some aspects of uncertainty in computational fluid dynamics results. J Fluids Eng 1991;113(4): 38–43.
[213] Mehta UB. Guide to credible computational fluid dynamics simulations. AIAA Paper No. 95-2225, 1995.
[214] Mehta UB. Guide to credible computer simulations of fluid flows. J Propulsion Power 1996;12(5):940–8.
[215] Meyer B. Object-oriented software construction, 2nd ed. Upper Saddle River, NJ: Prentice-Hall, 1997.
[216] Milliken GA, Johnson DE. Analysis of messy data, vol. 1: Designed experiments. Belmont, CA: Lifetime Learning Publications, 1984.
[217] Miser HJ. A foundational concept of science appropriate for validation in operational research. Eur J Oper Res 1993;66:204–15.
[218] Mittal R, Balachandar S. Effect of three-dimensionality on the lift and drag of nominally two-dimensional cylinders. Phys Fluids 1995;7(8):1841–65.
[219] Mittal R, Balachandar S. Vortical structures in bluff body wakes. AIAA-95-0867, 1995.
[220] NAFEMS. CFD working group, International Association for the Engineering Analysis Community, www.NAFEMS.org/cfdwg/cfdpage.html.
[221] Najjar FM, Vanka SP. Effects of intrinsic three-dimensionality on the drag characteristics of a normal flat plate. Phys Fluids 1995;7(10):2516–8.
[222] NASA. Computational fluid dynamics: codes, developments, and applications. NASA Langley Research Center, http://ad-www.larc.nasa.gov/tsab/cfdlarc/.
[223] Nayfeh AH. Perturbation methods. New York: Wiley, 1973.
[224] Naylor TH. Computer simulation experiments with models of economic systems. New York: Wiley, 1971.
[225] Naylor TH, Finger JM. Verification of computer simulation models. Manage Sci 1967;14(2):92–101.
[226] Neelamkavil F. Computer simulation and modelling, 1st ed. New York: Wiley, 1987.
[227] Neumann RD. CFD validation—the interaction of experimental capabilities and numerical computations. AIAA Paper No. 90-3030, 1990.
[228] Nishida H, Satofuka N. Higher-order solutions of square driven cavity flow using a variable-order multi-grid method. Int J Numer Methods Eng 1992;34(2): 637–53.
[229] Nordstrom J. Accurate solutions of the Navier–Stokes equations despite unknown outflow boundary data. J Comput Phys 1994;120:184–205.
[230] NPARC. CFD Verification & validation, NPARC Alliance, www.lerc.nasa.gov/www/wind/valid.
[231] NRC. Current capabilities and future directions in computational fluid dynamics. Washington, DC: National Research Council, 1986.
[232] NRC. Statistical software engineering, National Research Council. Washington, DC: National Academy Press, 1996.
[233] O'Connell E, Saiedian H. Can you trust software capability evaluations? Computer 2000;33(2):28–35.

[234] Oberkampf WL. A proposed framework for computational fluid dynamics code calibration/validation. AIAA Paper No. 94-2540, 1994.
[235] Oberkampf WL. Bibliography for verification and validation in computational simulation. SAND98-2041, Sandia National Laboratories, Albuquerque, NM, 1998.
[236] Oberkampf WL, Aeschliman DP. Joint computational/experimental aerodynamics research on a hypersonic vehicle: part 1, experimental results. AIAA J 1992; 30(8):2000–9.
[237] Oberkampf WL, Aeschliman DP, Henfling JF, Larson DE. Surface pressure measurements for CFD code validation in hypersonic flow. AIAA Paper No. 95-2273, 1995.
[238] Oberkampf WL, Aeschliman DP, Henfling JF, Larson DE, and Payne, JL. Surface pressure measurements on a hypersonic vehicle. AIAA Paper No. 96-0669, 1996.
[239] Oberkampf WL, Aeschliman DP, Tate RE, Henfling JF. Experimental aerodynamics research on a hypersonic vehicle. SAND92-1411, Sandia National Laboratories, Albuquerque, NM, 1993.
[240] Oberkampf WL, Blottner FG. Issues in computational fluid dynamics code verification and validation. AIAA J 1998;36(5):687–95.
[241] Oberkampf WL, Blottner FG, Aeschliman DP. Methodology for computational fluid dynamics code verification/validation. AIAA Paper No. 95-2226, 1995.
[242] Oberkampf WL, DeLand SM, Rutherford BM, Diegert KV, Alvin KF. A new methodology for the estimation of total uncertainty in computational simulation. AIAA Paper No. 99-1612, 1999.
[243] Oberkampf WL, DeLand SM, Rutherford BM, Diegert KV, Alvin KF. Estimation of total uncertainty in computational simulation. SAND2000-0824, Sandia National Laboratories, Albuquerque, NM, 2000.
[244] Oberkampf WL, Diegert KV, Alvin KF, Rutherford BM. Variability, uncertainty, and error in computational simulations, ASME-HTD-vol. 357-2. New York: American Society of Mechanical Engineers, 1998.
[245] Oberkampf WL, Martellucci A, Kaestner PC. SWERVE Surface pressure measurements at mach numbers 3 and 8. SAND84-2149, Sandia National Laboratories, SECRET Formerly Restricted Data, Albuquerque, NM, 1985.
[246] Oberkampf WL, Trucano TG. Validation methodology in computational fluid dynamics. AIAA 2000-2549, 2000.
[247] Oden JT, Feng Y, Prudhomme S. Local and pollution error estimation for Stokesian flow. Int J Numer Methods Fluids 1998;27:33–9.
[248] Oral M, Kettani O. The facets of the modeling and validation process in operations research. Eur J Oper Res 1993;66:216–34.
[249] Oren TI. Concepts and criteria to assess acceptability of simulation studies: a frame of reference. Commun ACM 1981;24(4):180–9.
[250] Oren TI, Zeigler BP, Elzas MS, editors. Simulation and model-based methodologies: an integrative view. Berlin: Springer, 1984.
[251] Oreskes N, Shrader-Frechette K, Belitz K. Verification, validation, and confirmation of numerical models in the earth sciences. Science 1994;263:641–6.
[252] Pace DK. Fidelity Considerations for RDE distributed simulation, 1. 1997 Fall Simulation Interoperability Workshop Papers 1997. p. 249–59.
[253] Pace DK. Dimensions and attributes of simulation fidelity, 1. 1998 Fall Simulation Interoperability Workshop Papers, 1998.
[254] Paciorri R, Dieudonne W, Degrez G, Charbonnier J-M, Deconinck H. Exploring the validity of the Spalart–Allmaras turbulence model for hypersonic flows. J Spacecraft Rockets 1998;35(2):121–6.
[255] Palmer TN. Predicting uncertainty in forecasts of weather and climate. Rep Prog Phys 2000;63:71–116.
[256] Panton RL. Incompressible flow. New York: Wiley, 1984.
[257] Parry GW. The characterization of uncertainty in probabilistic risk assessments of complex systems. Reliability Eng System Saf 1996;54:119–26.
[258] Partsch HA. Specification and transformation of programs. New York: Springer-Verlag, 1990.
[259] Paté-Cornell ME. Uncertainties in risk analysis: six levels of treatment. Reliability Eng System Saf 1996;54:95–111.
[260] Paulk MC, Weber CV, Curtis B, Chrissis MB, editors. The capability maturity model: guidelines for improving the software process. Reading, MA: Addison-Wesley, 1994.
[261] Peercy DE. Personal Communication, 2000.
[262] Pilch M, Trucano TG, Moya JL, Froehlich GK, Hodges AL, Peercy DE. Guidelines for Sandia ASCI verification and validation plans—content and format: Version 2. SAND2000-3101, Sandia National Laboratories, Albuquerque, NM, 2001.
[263] Popper KR. The logic of scientific discovery. New York: Basic Books, 1959.
[264] Popper KR. Conjectures and refutations: the growth of scientific knowledge. London: Routledge and Kegan, 1969.
[265] Porter JL. A summary/overview of selected computational fluid dynamics (CFD) code validation/calibration activities. AIAA Paper No. 96-2053, 1996.
[266] PSI. PureCoverage: User's guide. Sunnyvale, CA: Pure Software, Inc., 1994.
[267] Rai SN, Krewski D, Bartlett S. A general framework for the analysis of uncertainty and variability in risk assessment. Hum Ecol Risk Assess 1996;2(4):972–89.
[268] Red-Horse JR, Paez TL, Field RV, Romero V. Nondeterministic analysis of mechanical systems. SAND2000-0890, Sandia National Laboratories, Albuquerque, NM, 2000.
[269] Reed HL, Haynes TS, Saric WS. Computational fluid dynamics validation issues in transition modeling. AIAA J 1998;36(5):742–51.
[270] Richardson LF, Gaunt JA. The deferred approach to the limit. Trans R Soc London, Ser A: Math Phys Sci 1927;226:299–361.
[271] Richtmeyer RD, Morton KW. Difference methods for initial-value problems. New York, NY: Interscience, 1967.
[272] Rizzi A, Vos J. Toward establishing credibility in computational fluid dynamics simulations. AIAA J 1998;36(5):668–75.
[273] Roache PJ. Scaling of high-Reynolds-number weakly separated channel flows. In: Cebecci T, editor. Numerical

aspects of physical aspects of aerodynamic flows. New York: Springer-Verlag, 1982. p. 87–98.
[274] Roache PJ. Need for control of numerical accuracy. J Spacecraft Rockets 1990;27(2):98–102.
[275] Roache PJ. Perspective: a method for uniform reporting of grid refinement studies. J Fluids Eng 1994;116: 405–13.
[276] Roache PJ. Verification of Codes and Calculations. AIAA Paper No. 95-2224, 1995.
[277] Roache PJ. Quantification of uncertainty in computational fluid dynamics. In: Lumley JL, Van Dyke M, editors. Annual review of fluid mechanics. Palo Alto, CA: Annual Reviews, Inc., 1997. p. 126–60.
[278] Roache PJ. Verification and validation in computational science and engineering. Albuquerque, NM: Hermosa Publishers, 1998.
[279] Roache PJ. Verification of codes and calculations. AIAA J 1998;36(5):696–702.
[280] Roache PJ, Ghia KN, White FM. Editorial policy statement on the control of numerical accuracy. J Fluids Eng 1986;108(1):2.
[281] Roache PJ, Knupp PM. Completed Richardson extrapolation. Commun Numer Methods Eng 1993;9: 365–74.
[282] Roache PJ, Knupp PM, Steinberg S, Blaine RL. Experience with benchmark test cases for groundwater flow. In: Celik I, Freitas CJ, editors. Benchmark test cases for computational fluid dynamics, FED-vol. 93. New York: The American Society of Mechanical Engineers, 1990. p. 49–56.
[283] Rook P. Software reliability handbook. New York: Elsevier Science Publishers, 1990.
[284] Rowe WD. Understanding uncertainty. Risk Anal 1994; 14(5):743–50.
[285] Roy CJ, Blottner FB. Assessment of one- and two-equation turbulence models for hypersonic flows. J Spacecraft Rockets 2001;38(5):699–710.
[286] Roy CJ, McWherter-Payne MA, Oberkampf WL. Verification and validation for laminar hypersonic flowfields. AIAA2000-2550, 2000.
[287] Rushby J. Formal methods and the certification of critical systems. SRI-CSL-93-7, Computer Science Laboratory, SRI International, Menlo Park, CA, 1993.
[288] Salari K, Blaine RL, Economy K, Roache PJ. Grid resolution studies of radionuclide transport in fractured porous media, FED-213, Joint JSME-ASME Fluid Mechanics Meeting, Hilton Head, SC, 1995. p. 25–30.
[289] Salari K, Knupp P. Code verification by the method of manufactured solutions. SAND2000-1444, Sandia National Labs, Albuquerque, NM, 2000.
[290] Saltelli A, Scott M. Guest editorial: The role of sensitivity analysis in the corroboration of models and its link to model structural and parametric uncertainty. Reliability Eng System Saf 1997;57:1–4.
[291] Sanders F, Mullen SL, Baumhefner DP. Ensemble simulations of explosive cyclogenesis at ranges of 2–5 days. Mon Weather Rev 2000;128(8/pt. 2):2920–34.
[292] Sanders J, Curran E. Software quality. Reading, MA: Addison-Wesley, 1994.
[293] Sargent RG. Simulation model validation. In: Oren TI, Zeigler BP, Elzas MS, editors. Simulation and model-based methodologies: an integrative view. Berlin: Springer-Verlag, 1984. p. 537–55.
[294] Sargent RG. An expository on verification and validation of simulation models. 1985 Winter Simulation Conference, Sacramento, CA, 1985. p. 15–22.
[295] Sargent RG. Validation of mathematical models. Symposium on Validation of Geosphere Flow and Transport Models, Stockholm, Sweden, 1990. p. 571–9.
[296] Sargent RG. Verifying and validating simulation models. 1996 Winter Simulation Conference, Coronado, CA, 1996. p. 55–64.
[297] Schlesinger S. Terminology for model credibility. Simulation 1979;32(3):103–4.
[298] Schreiber F, Keller HB. Driven cavity flows by efficient numerical techniques. J Comput Phys 1983;49(2): 310–33.
[299] Serrano SE. Analytical solutions of the nonlinear groundwater flow equation in unconfined aquifers and the effect of heterogeneity. Water Resources Research 1995;31(11):2733–42.
[300] Settles GS, Dodson LJ. Hypersonic shock/boundary-layer interaction database. NASA, Contractor Rept. 177577, 1991.
[301] Settles GS, Dodson LJ. Hypersonic turbulent boundary-layer and free shear layer database. NASA, Contractor Rept. 177610, 1993.
[302] Settles GS, Dodson LJ. Supersonic and hypersonic shock/boundary-layer interaction database. AIAA J 1994;32(7):1377–83.
[303] Shafer G. A mathematical theory of evidence. Princeton, NJ: Princeton University Press, 1976.
[304] Shannon RE. Systems simulation: the art and science. Englewood Cliffs, NJ: Prentice-Hall, Inc., 1975.
[305] Sheng G, Elzas MS, Oren TI, Cronhjort BT. Model validation: a systemic and systematic approach. Reliability Eng System Saf 1993;42:247–59.
[306] Shih TM. A procedure to debug computer-programs. Int J Numer Methods Eng 1985;21(6):1027–37.
[307] Shih TM, Tan CH, Hwang BC. Effects of grid staggering on numerical schemes. Int J Numer Methods Fluids 1989;9:193–212.
[308] Shimazaki K, Himeno Y, Baba N. Quantification of uncertainty in computational fluid dynamics, FED-vol. 158, New York: The American Society of Mechanical Engineers, 1993.
[309] Shirazi SA, Truman CR. Evaluation of algebraic turbulence models for PNS predictions of supersonic flow past a sphere-cone. AIAA J 1989;27(5):560–8.
[310] Sindir MM, Barson SL, Chan DC, Lin WH. On the development and demonstration of a code validation process for industrial applications. AIAA Paper No. 96-2032, 1996.
[311] Sindir MM, Lynch ED. Overview of the state-of-practice of computational fluid dynamics in advanced propulsion system design. AIAA Paper No. 97-2124, 1997.
[312] Singhal AK. Validation of CFD codes and assessment of CFD simulations. Fourth International Symposium of Transport Phenomena and Dynamics of Rotating Machinery. Honolulu, HI, 1992. p. 155–169.
[313] Singhal AK. Key elements of verification and validation of CFD software. AIAA 98-2639, 1998.

[314] Slater JW, Dudek JC, Tatum KE. The NPARC alliance verification and validation archive. ASME 2000-FED-11233, 2000.
[315] Smith S, Kandel A. Verification and validation of rule-based expert systems. Boca Raton, FL: CRC Press, 1993.
[316] Smithson M. Ignorance and uncertainty: emerging paradigms. New York: Springer-Verlag, 1989.
[317] Springer AM. Comparison of the aerodynamic characteristics of similar models in two different size wind tunnels at transonic speeds. AIAA-98-2875, 1998.
[318] Srinivasan R. Accurate solutions for steady plane flow in the driven cavity. I. Stokes flow. Z Angew Math Phys 1995;46(4):524–45.
[319] Srivastava BN, Werle MJ, Davis RT. A finite difference technique involving discontinuous derivatives. Comput and Fluids 1979;7(1):69–74.
[320] Steinberg S, Roache PJ. Symbolic manipulation and computational fluid dynamics. J Comput Phys 1985; 57(2):251–84.
[321] Stevenson DE. A critical look at quality in large-scale simulations. Comput Sci Eng 1999;1(3):53–63.
[322] Stockman CT, Garner JW, Helton JC, Johnson JD, Shinta A, Smith LN. Radionuclide transport in the vicinity of the repository and associated complementary cumulative distribution functions in the 1996 performance assessment for the waste isolation pilot plant. Reliability Eng System Saf 2000;69(1–3):369–96.
[323] Sudicky EA, Frind EO. Contaminant transport in fractured porous media: analytical solutions for a system of parallel fractures. Water Resour Res 1982;18(6): 1634–42.
[324] Summa JM, Barton JM. CFD verification and validation in commercial design and analysis. AIAA 98-2640, 1998.
[325] Sundaresan S, Nagarajan S, Deshpande SM, Narasimha R. 2D lid-driven cavity flow at high Reynolds numbers: some interesting fluid-dynamical issues. Sixteenth International Conference on Numerical Methods in Fluid Dynamics. Arcachon, France, 1998. p. 231–6.
[326] Sung CH, Fu TC, Griffin MJ, Huang TT. Validation of incompressible flow computation of forces and moments on axisymmetric bodies at incidence. AIAA-95-0528, 1995.
[327] Terrill RM, Colgan T. Some simple analytic solutions of the Navier–Stokes equations. Int J Eng Sci 1991;29(1): 55–68.
[328] Tsang C-F. A broad view of model validation. Proceedings of the Symposium on Safety Assessment of Radioactive Waste Repositories. Paris, France, 1989. p. 707–16.
[329] Van De Vooren AI, Dijkstra D. The Navier–Stokes solution for laminar flow past a semi-infinite flat plate. J Eng Math 1970;4(1):9–27.
[330] Van Wie DM, Rice T. Quantification of data uncertainties and validation of CFD results in the development of hypersonic airbreathing engines. AIAA 96-2028, 1996.
[331] Veazey DT, Hopf JC. Comparison of aerodynamic data obtained in the Arnold Engineering Development Center Wind Tunnels 4T and 16T. AIAA 98-2874, 1998.
[332] Venditti DA, Darmofal DL. Adjoint error estimation and grid adaptation for functional outputs: application to quasi-one-dimensional flow. J Comput Phys 2000; 164(1):204–27.
[333] Venkateswaran S, Merkle CL. Evaluation of artificial dissipation models and their relationship to the accuracy of Euler and Navier–Stokes computations, 515. Sixteenth International Conference on Numerical Methods in Fluid Dynamics, Arcachon, France, 1998. p. 427–32.
[334] Verhoff A. Far-field computational boundary conditions for three-dimensional external flow problems. AIAA-96-0892, 1996.
[335] Verhoff A. Complementing numerical simulation methods with classical analytical techniques. AIAA-98-2486, 1998.
[336] Verhoff A, Cary A. Analytical Euler solutions for 2D flows with corners using asymptotic methods. AIAA-98-2687, 1998.
[337] Wahlbin LB. Numerical analysis on non-smooth problems: some examples. In: Dahlberg B et al., editors. Partial differential equations with minimal smoothness and applications. New York, NY: Springer-Verlag, 1992. p. 213–20.
[338] Walker MA, Oberkampf WL. Joint computational/experimental aerodynamics research on a hypersonic vehicle: part 2, computational results. AIAA J 1992;30(8):2010–6.
[339] Wallace DR, Ippolito LM, Cuthill BB. Reference information for the software verification and validation process. Rept. 500–234, 1996.
[340] Wang CY. Exact solutions of the steady-state Navier–Stokes equations. In: Lumley JL, Van Dyke M, editors. Annual review of fluid mechanics. Palo Alto, CA: Annual Reviews, Inc., 1991. p. 159–77.
[341] White FM. Viscous fluid flow. New York: McGraw Hill, 1991.
[342] Widmann JF, Charagundla SR, Presser C, Heckert A. Benchmark experimental database for multiphase combustion model input and validation: baseline case. NIST, US Department of Commerce, NISTIR 6286, Gaithersburg, MD, 1999.
[343] Wilcox DC. Perturbation methods in the computer age. La Canada, CA: DCW Industries, 1995.
[344] Wilson GE, Boyack BE. The role of the PIRT in experiments, code development and code applications associated with reactor safety assessment. Nucl Eng Des 1998;186:23–37.
[345] Wise JA, Hopkin VD, Stager P, editors. Verification and validation of complex systems: human factors issues. Berlin: Springer-Verlag, 1993.
[346] Yee HC, Sweby PK. Aspects of numerical uncertainties in time marching to steady-state numerical solutions. AIAA J 1998;36(5):712–24.
[347] Yoshizawa A. Laminar viscous flow past a semi-infinite flat plate. J Phys Soc Japan 1970;28(3):776–9.
[348] Youden WJ. Enduring values. Technometrics 1972;14(1): 1–11.
[349] Zeigler BP. Theory of modelling and simulation, 1st ed. New York: Wiley, 1976.
[350] Zhang XD, Pelletier D, Trepanier JY, Camarero R. Verification of error estimators for the Euler equations. AIAA-2000-1001, 2000.

[351] Zhang XD, Trepanier JY, Camarero R. An a posteriori error estimation method based on error equations. AIAA-97-1889, 1997.

[352] Zimmerman DA, deMarsily G, Gotway CA, Marietta MG, Axness CL, Beauheim RL, Bras RL, Carrera J, Dagan G, Davies PB, Gallegos DP, Galli A, GomezHernandez J, Grindrod P, Gutjahr AL, et al. A comparison of seven geostatistically based inverse approaches to estimate transmissivities for modeling advection transport by groundwater flow. Water Resour Res 1998; 34(6):1373–413.

[353] Zingg DW. Grid studies for thin-layer Navier–Stokes computations of airfoil flowfields. AIAA J 1992; 30(10):2561–4.