

The Role of Educational Institutions in Verification and Validation Preparation

J. Desel

Lehrstuhl für Angewandte Informatik
Katholische Universität Eichstätt
85071 Eichstätt, GERMANY

R. Panoff

Shodor Education Foundation
923 Broad Street, Suite 100
Rayleigh, NC 27705, USA

P. Gray

Department of Computer Science
University of Northern Iowa
Cedar Falls, IA 50614-0507, USA

D. E. Stevenson

Department of Computer Science
Clemson University
Clemson, SC 29634-1906, USA

Abstract

This paper presents a snapshot of the current trends in academic coverage of verification and validation (V&V) principles in engineering and the computational, mathematical, and physical sciences. While a well-established component of modeling and simulation in government programs and in industry, the importance of V&V has not yet permeated the curriculum of our educational institutions. Educational programs could play a far more significant role in this field by emphasizing how to identify and prevent common errors in the modeling process.

Suggestions are given on integrating the concepts of V&V into the curriculum in the form of topics-based modules, on moving these concepts into the core of the curriculum, and on restructuring academic programs so as to position the aspects of V&V in modeling and simulation as fundamental components of the educational experience. Recommendations for the role of cross-cutting programs and cross-disciplinary programs in V&V education are also stated.

1. Introduction

Verification and validation (V&V) has always been associated with very large-scale modeling and simulation. In fact, V&V plays such an integral role, that the importance of V&V training is now coming to the forefront of education. This paper addresses the role of

educational institutions in teaching solid V&V principles.

The scope and scale of modeling and simulation continues to push the boundaries of computational capabilities. Not only is this increasingly true in academic research (grid computing) and in long-range government projects (nuclear weapons testing), but in industry as well (computational fluid dynamics, bioinformatics). In order to support this trend and to produce graduates that are acclimated to the size of problems that will confront us in the generations to come, the status quo of academic programs will require a significant paradigm shift.

In recent history, educational institutions have shown considerable flexibility in the face of changing times. Paradigm shifts in computer science from core languages ranging from PL-1, FORTRAN and COBOL, to C, to C++, and now toward Java is one example of academia changing with public and industrial trends. The adoption of the “Harvard calculus” series in mathematics is another. The appropriate and judicious infusion of V&V into the curriculum, however, will require full *interdisciplinary* efforts and considerable revamping of the educational core.

In general, verification and validation are terms that have been stigmatized in the eyes of academia. These terms have become intimately tied to very large-scale simulations and projects. As a consequence, they have become almost exclusively aligned with “government” or “industrial” endeavors.

There is a distinct separation between modeling and

simulation practices (M&S) and the omission of verification and validation (V&V) practices at the educational level. The impact of this omission of verification and validation components can be seen throughout history, and resound throughout the headlines today. While the foundations for modeling and simulation are rooted in mathematics, engineering, computer science and the physical sciences, the educational foundations for verification and validation are not nearly as deep nor as broad.

This creates a significant chasm between the type of researcher being produced by academia and the expectations for employers for which V&V is the norm. The importance of V&V in large-scale simulation is abundantly clear. That is to say, that industrial and governmental entities that develop large-scale simulations cannot be expected to lower the bar to cater to the lacking presence of V&V principles found in academia. The question then becomes “how should academia raise the overall presence and quality of V&V practices?”

As depicted in Section 3, V&V problems are ubiquitous and often embody extraordinary costs. Bolstering the presence and significance of V&V in education would have considerable returns. Our assertion is that the appropriate niche for V&V education lies primarily in state-transition systems and in determining if the defined system(s) reflect observations, at least in the statistical sense. In order to realize this objective, there must be greater pedagogical emphasis on

1. identification of errors that invalidate our modeling process.
2. analysis using ordinary differential equations and elementary extensions to partial differential equation systems.
3. elementary stochastics and probability analysis.
4. numerical methods.
5. understanding stochastic processes.

In many ways, the first item encompasses many aspects of the others. As will be argued in Section 6.1, striving to educate students on how to identify and exorcize errors from the M&S process will produce the largest payoff to the V&V community.

Educating students on recognizing errant models and computational results can not be done in a single lecture, nor in a single course. (Nor would we argue that it should be attempted in this manner.) Rather, identifying and addressing the causes of errors in our models should be presented far more pervasively and in the context of the curriculum. In Section 5, we argue that coverage of V&V in the curriculum should resemble the process of “iterative-refinement:” when errors are discovered (or even suspected), the model should be—to the extent possible—reformulated to address the presence of the errors, and the consequences examined.

2. Varying Definitions With a Common Theme

There are many connotations and definitions for verification and validation (which only compounds the issue for academia). Depending upon the context, the definitions for both validation and verification differ significantly. Unfortunately, some of these definitions are not well suited for “educational” purposes. Different connotations exist based upon the application of V&V practices to support modeling and simulation, deliverables-driven production, or program accreditation. For example, the IEEE defines verification and validation with strong deliverables slant[12]:

“Verification is the process of evaluating a system or component to determine whether the products of a given phase satisfy the conditions imposed at the start of that phase”

“Validation is the process of evaluating a system or component during or at the end of the development process to determine whether it satisfies a specifies specified requirements.”

These definitions are very hard to mold into an academic setting. Implicit in these definitions is the idea of a phase-based development cycle which is hard to fit into traditional topics-based educational approaches outside of software development and engineering. Another reason why this definition is difficult to press into the academic mold is the assertion that the validation process occurs at the end of the development process. In an classroom or educational laboratory setting, validation issues would more naturally be presented as being relevant to individual topics, or with every stage of development.

The Department of Defense—being more concerned with accreditation and long-term projects with clearly-specified milestones—has somewhat orthogonal definitions. The DoD definitions and the definitions adopted by the AIAA ([1]) are similar:

“(Verification is the ...) process of determining that a model implementation accurately represents the developer’s conceptual description and specifications.”

“(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 model’s intended uses.”

While these definitions play more true to a modeling and simulation definition for V&V, these definitions require one to divine the conceptual components of the developer and to understand the intentions of the model’s use. If the intentions are simply for educational exercise, then the point of modeling becomes truly academic.

The “fallback” definitions for V&V thus become a de-facto standard for academia:

Verification addresses the question of “is the model right.” Validation addresses the question of “is it the right model.”

Appropriately, these definitions allow academia to focus on V&V issues surrounding modeling and simulation. A fundamental issue with the adoption of the previous definitions for V&V in an academic setting is that of a (falsely) perceived serialization of the modeling process. V&V fundamentals in the modeling and simulation educational process must be present in conceptualization through the derivation of computational solutions. It will be argued in sections 5.1 and 5.2, that the role of validation and verification in the educational modeling process is more representative of a refined, iterative process than a process to mark the end of an epoch.

The modeling and simulation process involves several stages that typically consist of five stages([16]):

1. conceptual modeling of a real system
2. mathematical modeling of the conceptual models
3. discretization and the judicious choice of algorithms
4. formulation of a derived or numerical solution and

5. representation of the derived or numerical solution.

Verification and validation add critical information to the process as a whole and apply equally to each individual step. Verification and validation broadens the entire modeling and simulation process by adding knowledge of our model’s boundaries, and by providing bounds to the realm of our ignorance. Ignorance, as defined by Ayyub in [3], is an omnipresent consideration in every model, brought about from errors, incompleteness, anomalies, irrelevance, inaccuracy, vagueness, probability, ambiguity, randomness, the model, conflict, and numerous other attributes that cannot be isolated easily in modeling “real-life” simulations.

There are many areas in which academia does very respectably bringing together M&S with the appropriate issues in V&V. For example, V&V *modules* appear in the curriculum in the forms of error analysis in a course in numerical analysis, error propagation analyses in physical chemistry and application stress testing in software engineering. Yet these modules are few and far between. They do not join together to form a logical V&V foundation. These modules do not address the larger-scale issue of the omission of V&V practices at the *curriculum* and *program* levels. Further, these types of V&V modules are not presented in light of their interdisciplinary significance or in establishing the soundness of the model.

Although a somewhat anecdotal benchmark, searching the top American and European academic textbook publishers for textbooks covering “Verification and Validation” is somewhat illuminating. A search of ten top academic publishers during the summer of 2002 returned less than ten books in print¹ that discussed verification and/or validation within the chapters. Only two of the books that resulted from the search specifically dealt with validation and verification as pertaining to modeling and simulation ([19],[9]). The majority of results were in the fields of engineering and software development².

¹One textbook that had a chapter devoted to Verification and Validation was no longer in print.

²The software development titles slanted almost exclusively toward verification.

3. Is this really a problem?

There are many headlines from past and recent history that would indicate the importance of V&V practices:

- The Mars Surveyor '98 program consisted of two separately-launched spacecraft: The Mars Climate Orbiter and the Mars Polar Lander. A navigation error caused the Mars Climate Orbiter to be destroyed when it missed its target altitude of 140-150 km. The actual altitude was estimated to be about 57km. The cause of the discrepancy was found by the failure review board to be commands sent in English units instead of being converted to metric[13]. The Lander met with a similar demise, attributable to a precondition violation in software, making the lander incorrectly believe that it had landed and to shutdown its engines[20].
- Ever present these days is the issue of computer security. According to CERT[6], the vast majority of security attacks exploit
 - input validation failures
 - buffer overflows
 - or the targeting of security/cryptographic software
- The crew of the USS Yorktown was dead in the water for more than two hours because a crew member mistakenly entered a zero into the data field of a Microsoft Windows NT application. The computer system proceeded to propagate the erroneous entry throughout the system by dividing other quantities by that zero. The operation caused a buffer overflow, in which data leaked from a temporary storage space in memory. The error eventually brought down the ship's propulsion system [15].
- After testing the algorithm, the program, the compiler and the motherboard, Thomas Nicely of Lynchburg College, VA. ruled out everything but the CPU as the cause of glitches in the computational results of his twin-prime research. Five missing entries in the Intel Pentium's lookup table that formed the circuitry for the radix-4 SRT algorithm was significant enough to throw off the

precision of Professor Nicely calculations. The radix-4 SRT algorithm provides the core of the Pentium's floating point division instruction[14].

- After 10 years and \$7 billion, the European Space Agency had all but cornered the commercial space business with the a giant rocket that was capable of hurling a pair of three-ton satellites into orbit with each launch. 40 seconds into its maiden flight on June 4, 1996, the Ariane 5 veered off course and exploded. Later, the cause was tracked down and attributed to an error in the reuse of a software component. The program tried to stuff a 64-bit number into a 16-bit space [8]. The failed guidance system transferred control to the secondary guidance system, which failed in an identical manner because it was running the same software.
- In January 1999, Lawrence Berkeley National Laboratory announced the discovery of the "superheavy" element 118 in a paper in *Physical Review Letters*. After a thorough analysis of the original data using different software codes, there was sufficient evidence to merit the public retraction of the discovery, published in the September 2002 edition of *Physical Review Letters*[17].
- Between May 4 and May 19, 2000, the Cerro Grande/Los Alamos fire destroyed the homes of more than 18,000 residents and inflicted damage estimated at about \$1 billion. The tragedy was the result of a prescribed fire ignited by officials of the National Park Service, whose intentions were, ironically, to reduce the risk of this very type of fire. In the end, the go/no-go decision was made based upon a burn simulation whose output indicated safe conditions, but with a variance that also included the significant plausibility of unsafe conditions[10].

Numerous other examples abound in literature (see, for example, [18]). As the above list illustrates, issues surrounding V&V practices are commonplace. Based upon the costs and potential for devastation illustrated in these examples, rigorous implementation of V&V principles are a cost-effective insurance policy.

4. Verification and Validation Standards

In general, Universities have blinders on when it comes to integrating techniques for validating and verifying models from separate disciplines. Is the issue a lack of standards? For example, appropriate selections from the 77 techniques for validation and verification of conventional modeling and simulation given by Osman Balci, [4], should be identified and highlighted throughout the curriculum. Yet opportunities for formal coverage remain largely unaddressed or glossed over at best. Taking an analysis of informal algorithmic models using Turing Tests and walkthroughs; a V&V analysis of static models through cause-effect graphing, state transition analysis, and semantic analysis; the verification and validation of dynamic models using beta testing, real-time input testing, and security testing; and the analysis of formal models using logical deduction, inference, and predicate calculus tools are all common techniques enumerated by Balci [4]. However, these common classroom techniques are not explicitly linked to V&V in the course of academic instruction.

This is consistent with the exploratory findings of the authors. While preparing this document, questionnaires concerning V&V in the curriculum were distributed to institutions of higher learning. Of the 175 surveys that were sent out, only 5 were returned; and of those 5 returned surveys, V&V was not represented well in the curriculum.

5. Validation and Verification in Academics

Both industrial and governmental organizations weigh the cost of V&V in the development cycle. Costs must be balanced with the acceptable level of confidence in the final product. Such a cost analysis has not been undertaken in academia, for it would clearly show the cost effectiveness of increasing the presence of V&V principles in the academic life cycle.

The “cost” of V&V in academics is time. Time would be expended on curriculum development and major restructuring of programs. Academic programs are currently top-heavy for depth, not breadth. Focus is introspective instead of on the overall interdependencies between the disciplines that exists in large-scale

modeling and simulation.

The adoption of V&V practices should be at all levels of instruction in the natural sciences. This includes development and incorporation of V&V-specific modules for individual courses, the inclusion of V&V practices as an underlying foundation for course work, through to the fundamental framework at the program level.

Throughout this section, the implementation of academic-centric V&V topics will be based upon the model provided by the AIAA[1]. The general interrelationship between the conceptual mode, the computerized model and reality is shown in Figure 1.

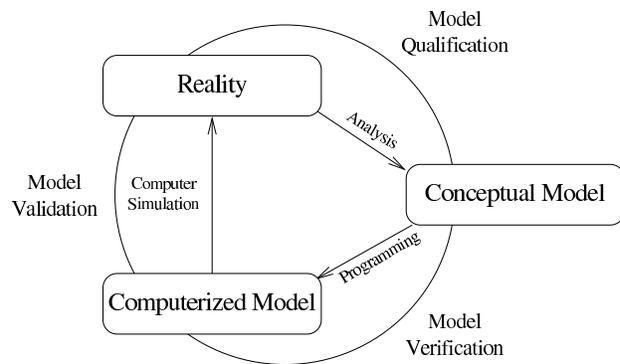


Figure 1. The AIAA relationship between the Conceptual Model, the Computerized model, and Reality.

In the following sections, categories where V&V principles can be leveraged upon are identified at various levels. These categories include modules, curriculum, and programs. Modules refer to in-class coverage of topics; curriculum refers to course content and the appropriate placement of material along the the course work track leading to a degree; a program is the collection of required courses prescribed to fulfill a students degree requirements.

5.1 V&V at the module level

Educational institutions should look to infuse the current curriculum in ways that raise the general awareness of V&V practices where appropriate. When aspects of “rapid application development” overshadow characteristics of program correctness and ro-

bustness, learning becomes top-heavy and lacks insight which is required for teaching students *intuition*.

One trend in computer science is to focus on presenting topics in a “top-down approach.” When presented without the components of verification, the result is a top-heavy approach. Verification should not be an “afterthought” in the development of bug-free programs, but rather integrated into each of the five basic principles of structured programming: *a top-down approach, modularity, compact modules, stepwise refinement, and structured control* [11].

Course modules are needed to bring the supporting role of V&V to the forefront. One approach would be for instructors to explicitly identify strategic V&V practices based upon the 77 categories for conventional simulation models and the 38 categories for object-oriented simulation models as detailed in [4]. This would place minimal demands on the current curriculum as these topics are already present implicitly in the subject material; explicit coverage in a modular form would (a) lend toward awareness (b) establish the foundation for more detailed V&V modules built upon several topics, and (c) provide a spring board for spawning courses and programs devoted entirely to V&V practices within M&S.

In this manner, V&V “modules” would serve to reinforce the stated relationship between modeling, computation, and the verification against correct or accurate solutions. Modules would provide a means to scrutinize the output from computational methods, provide for reassessment of discretization or algorithm selection, and provide a means for reformulating the model. In this way, modules would be refinement links that provide for iteration amongst the AIAA components of verification, as shown in Figure 2.

V&V modules would play a strong supporting role in model development, model analysis, simulations, and application engineering. In an academic situation, where “delivery” of a model ranges from homework submissions through topic examination, V&V provides the transition between building and delivering the final product. This relationship is also illustrated in Figure 2. When viewed in this manner, the “spiral” interconnection of design stages is closely aligned with the “iterative-refinement” cycle adopted for product delivery in industry.

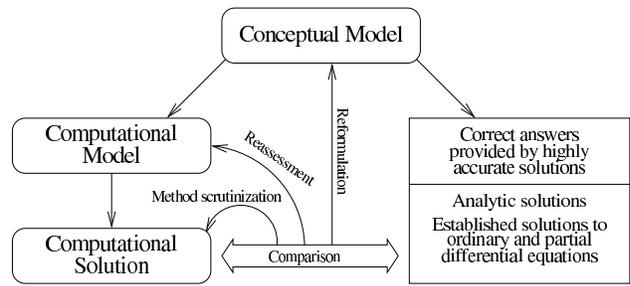


Figure 2. V&V education in the form of supporting materials (modules) would serve to provide “iterative refinement” for the AIAA validation process.

5.2 V&V at the curriculum level

Building upon the spiral reinforcement of V&V at the module level, the curriculum level should provide for an evolutionary development, analogous to the evolutionary V&V process given in [5]. Subject material will naturally require an underlying V&V framework—in the form of modules as mentioned above, for example—but will also require an upward integration path into the program level.

By building upon a framework of V&V modules, educational institutions would be able to support courses with underlying *validation-built-upon-modules* themes. For example, having laid the foundation using modules focusing on *debugging, white-box testing, performance testing, and assertion checking*, a course in software design would be able to address issues surrounding validation (and perhaps accreditation) at the culmination of the course.

In this way, a V&V-enriched curriculum supports evolutionary development of V&V principles, iterating from the fine-grain V&V presence from the integration of smaller modules through the course-wide objectives that bind the modules together for the validation picture. In addressing the curriculum in this manner, institutions provide the course-level interconnection of the validation cycle outlined by AIAA, as shown in Figure 3.

In mathematics, for example, a first course in differential equations might address various growth models, such as constant growth, linear growth, and logarithmic models such as Gompertz’ and Verhulst’s equa-

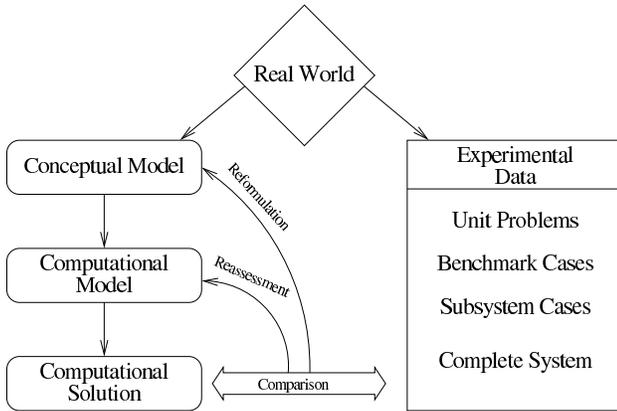


Figure 3. V&V evolution at the curriculum level would also serve to support refinement and improvement of the modeling process.

tions. The traditional textbook presentation develops each model very systematically, but commonly fails to bridge the validation chain as indicated in Figure 3. Individual presentations on each of these models in isolation fails to illustrate how to bring together real-world data and the modeling process. Presenting each model independently fails to identify the appropriateness of the model, giving insight into reformulation and fails to show how to scrutinize a computational solution to a more advanced model that no longer contains an easily-obtained analytical solution.

On the other hand, an example of a curricula’s embodiment of this paradigm is found in [7]. A course focusing on simulations as a means for model validation has been built upon modules composed of the tools, objects, and content of V&V. The overriding topic of the curriculum presented in [7], Petri nets, was shown to be a valid approach to simulation education in general, and distance learning in particular.

5.3 V&V at the program level

At the program level, having interdisciplinary content will be necessary in order to fundamentally support the significance of V&V. The intermingling of mathematical, scientific and computational epistemologies will also be required in order to address the high-performance computing and grand challenge issues that face us today.

Typical academic programs consist of very linear course progressions. A typical core program in computer science is illustrated in Figure 4, where course dependencies and hierarchies are narrowly focused. Such a layout sacrifices breadth of education at the expense of depth. This is *not* to argue that current programs have too much depth, but rather to emphasize that insight and ideas are reinforced by seeing context to their implementation.

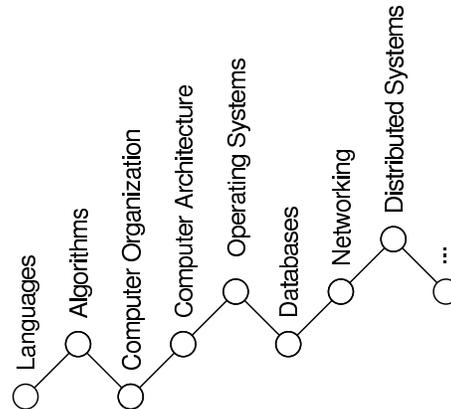


Figure 4. The current academic focus is very linear, and the broad understanding required for V&V is completely overlooked.

The ideal program would have both new programs building upon the core—as is traditionally done—as well as the integration of courses that pull together the fundamental ideas with interdisciplinary impact. This extends the traditional programmatic view of Figure 4 into a more *full multigrid* program structure, as depicted in Figure 5.

Admittedly, this would require a paradigm shift beyond what most departments and institutions would be willing to support. Nonetheless, in theory, a program managed by the involvement of several interdisciplinary departments would be able to support such an approach.

6. Identifying Errors: The Crux of V&V Education

There are many fundamental ways in which V&V techniques are already integrated into the educational process. At the grade school level, addition is shown

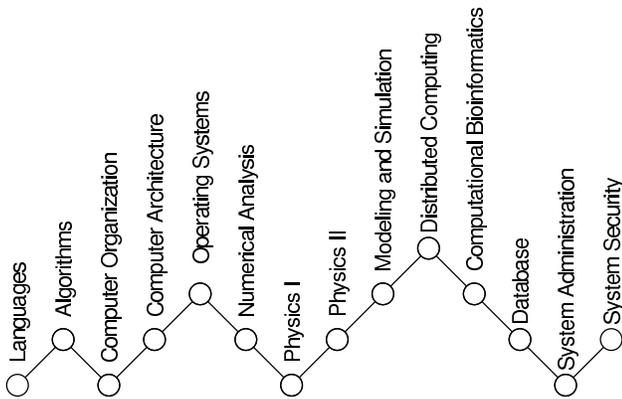


Figure 5. The ideal academic focus would provide an interdisciplinary perspective and would provide both top-down and bottom-up integration of new materials.

to be a very powerful verification technique for subtraction; multiplication is an equally important verification technique for division. Even counting fingers to add numbers is an application of model validation.

At the undergraduate level, verification takes on obvious forms; differentiation is used as a verification tool for integration techniques and solutions for differential equations; in linear algebra, solutions to the linear equations $x = A^{-1}b$ are verified by examination of the residual $r = Ax - b$.

Where undergraduate education largely fails is in validation: in addressing the limits of where model equations are no longer valid. In doing so, students never learn the right questions to ask.

Undergraduate education in the natural sciences is largely content to remain within arbitrary closed-world models. It has become standard practice to present a simplified model as if it completely encapsulates the real world, to analyze the equations and components, and to come away with the impression that the phenomena is completely understood. To a large extent, the burden of exposing students to issues involving model validation fall upon statistics and numerical analysis. But coverage needs to be significantly broadened to include a thorough examination of where errors can creep into the modeling process.

6.1 Error identification

Why is it that $\sin^2(x) + \cos^2(x) = 1$, a perfectly valid analytical expression, is only computationally true 97% of the time for angles between 0° and 45° ? Where did the error come from? What repercussions would this have on a computational solution to a mathematical model that depends upon this “analytical truth?” In order to validate a model, one must be able to identify and plug the holes that allow errors to enter into the modeling process.

One significant disservice to undergraduate education lies largely in the omission of *error identification*. Too often, mathematical models are presented, analyzed, and solved without a discussion of the weaknesses of the model or its solution.

Academic programs are accustomed to instruction of canned solutions to simplified problems as opposed to instruction in problem solving. In order to teach problem solving viz-a-viz modeling and simulation, students must be given a firm grasp on the errors that they will encounter during the process. Showing students how to identify sources of errors is fundamental to the avoidance of problems such as those listed in Section 3. Consider the categories of errors, listed below, as they relate to the manner in which modeling is presented in the undergraduate curriculum. Certain error categories are addressed in courses in numerical analysis (see [2] for example), software engineering, and the calculus sequences. However the overall importance and *interrelationships* between these errors are overlooked throughout all the sciences.

Errors in the mathematical equations used to represent physical reality. Mathematical equations are tools that are used to represent real world phenomena. The act of applying mathematics to emulate physical phenomena often marks the beginning of the modeling process. Errors introduced at this stage cannot be overcome.

To illustrate this concept, consider the modeling of a simple pendulum. The classic derivations of the model involve an analysis of a mass m attached to a string of length l , displaced slightly from its equilibrium. Two tradition approaches to deriving the mathematical model involve examination of either the forces acting upon the mass m , or in terms of rotational torque.

The torque on the mass is expressed as

$$\tau = mgl \sin \theta.$$

This equation is only an approximation of the torque which uses an approximation to gravity and neglects force components attributable to motion. But too often the derivation continues along uncontested. The next step would be to equate torque with the rotational analog of Newton's law, Ia , or

$$I \frac{d^2\theta}{dt^2} = -mgl \sin(\theta).$$

The model is simplified further under the assumption that $\sin(\theta) \sim \theta$ for small θ . Hence

$$I \frac{d^2\theta}{dt^2} = -mgl\theta.$$

This second-order differential equation in θ can be solved to produce the classic simple pendulum model, where the frequency and period can be reduced to the mass-free expressions

$$\omega = \sqrt{\frac{g}{l}} \quad T = 2\pi\sqrt{\frac{l}{g}}.$$

This would lead one to believe that the motion of a physical pendulum is sinusoidal, which it is not (but the *model* is). Where does the model go wrong? It fails right from the onset, where assumptions and simplifications are made so as to allow analytical solvability.

Blunders, goofs, mistakes. Prior to the widespread use of computers, arithmetic mistakes were the most common forms of errors categorized as “blunders.” Today, blunders are more likely attributable to programming errors during the implementation of algorithms and in general program correctness. To detect such errors, it is important for undergraduate programs to emphasize ways to scrutinize the accuracy of program output. If possible, programs should mimic cases where known, correct solutions exist. With trends toward rapid application development, code reuse in object-oriented programming environments, and the growing size and complexity in application programs, testing of each individual component is critical. Smaller portions of code should be tested independently. When complex programs are believed to run correctly, they should be heavily scrutinized under numerous usage scenarios.

Errors in Safe Coding. This category of errors encompasses many errors due to a lack of understanding of how computer systems “work.” These errors involve problems arising from overlaying program execution on top of system implementations. These are errors where the algorithm may be sound but the implementation of the algorithm is flawed. Examples are program implementations that overlook loss of significance errors, errors in computer representation of numbers, underflow and overflow errors, nested multiplication for polynomial evaluation (Horner's method), memory leaks, and lack of input validation. To a large extent, these errors are manageable if the application programmer has a fundamental understanding of how the interface between the program and the execution of the program is implemented.

When the use of `gets()` for user input is still supported and simple code fragments such as

```
if(10 * 0.1 == 1) {
```

result in undesired application behavior, this becomes far more than a numerical analysis issue.

With increasingly-complex programs, application programmers are encouraged to focus on the high-level aspects of code design and to forgo testing for system influences. Users have been involuntarily promoted to beta tester status as a result. Debugging and safe coding practices have been demoted in priority. Bounds checking, input validation, permissions verification, mutex protection, semaphore locking and memory management have been relegated to help-desk issues and service pack updates.

Parsing the `gcc` man pages, there are roughly thirty optimizations options that will strive to make programs run quicker or to produce smaller executables. There are no apparent compiler optimization options to make code run more stable, reliable or safe. The Ada programming language has an untarnished reputation as the appropriate language to use in embedded and real-time systems; should Ada have a more prominent role in the undergraduate curriculum? Using Java as a core programming language in computer science has many educational benefits. But despite the popularity and growth of Java, C and C++ remain entrenched as languages that are commonly used for modeling and for traditional high-performance appli-

cations. Are students equipped to make the transition to C or C++ once out of the safety of the Java Virtual Machine? The educational aspects associated with the use of memory profilers, code verifiers, and debuggers has waned in lieu of the rapid prototyping and code reuse paradigms.

Error in data collection. Many modeling problems arise from physically-observed phenomena. Recorded observations contain observational and instrumentation errors. To compensate for (or exacerbate) these errors, curve fittings, interpolations, and extrapolations are applied to the data.

Because there is error in the collection and recording of physical data, models based upon the data will have inherent limitations. For example, depending upon the meteorological model, the temperature data used, the curve fit, the extrapolation, etc., the earth's climate is either in a global warming phase or an impending ice age.

While these errors cannot be removed from the input data, an interval analysis, error propagation analysis, and probability analysis can be used to assess the impact of the propagation effects of these errors. While these are standard analysis tools used for verification, their significance needs to be emphasized considerably more in the undergraduate curriculum.

Numerical algorithm errors. These errors are the main focus of any undergraduate course in numerical analysis. These errors arise when one is confronted with a problem that cannot be solved analytically. In that case, one must resort to approximation methods.

Canonical examples are the evaluation of $\pi = 4 \tan^{-1}(1)$ and evaluating the integral

$$\int_0^1 e^{x^2} dx,$$

which has no anti-derivative. The standard approach—in an academic sense—for approximating these values derives from Taylor approximations, with verification through an integral analysis or bounding of the truncation error.

Taylor approximations maintain a strong presence in the undergraduate calculus courses, yet Taylor's remainder formula (the expression for truncation error)

is quickly losing ground in textbook coverage. In doing so, we run the risks associated with showing how to approximate expressions without showing how to reconcile good approximations from bad ones. For example, a common topic in the calculus curriculum is to derive the approximation

$$\cos(x) \approx 1 - \frac{x^2}{2} + \frac{x^4}{4!} - \dots + (-1)^n \frac{x^{2n}}{(2n)!}.$$

Often missing from this presentation to students is how to bound the approximation error using an analysis of the remainder formula or even why—due to loss-of-significance and the magnitudes of the computational errors—this is not a very good way to approximate $\cos(x)$ for $x \approx 2\pi$.

7. Addressing these issues

The previous sections set up many scenarios where M&S and V&V education could be improved upon in the undergraduate curriculum. By broadening students' exposure to errors that arise in the modeling process, by adopting a standard for pedagogical V&V modules, by augmenting curriculum, and by strengthening programs, the educational aspects of V&V training would be considerably strengthened. In many ways, each of the categories of errors described in the previous section are addressed in one form or another within various existing courses. For example, the analysis of the truncation error in Taylor approximations is a prime example of a V&V *module* that is currently (or at least should be) part of the standard calculus curriculum, and is in line with the module descriptions presented in Section 5. The issues surrounding loss-of-significance and nested polynomial multiplications are two prime examples of V&V modules for the computational sciences curricula.

The missing components involve cross-disciplinary understandings and interrelationships between these topics. In order to support undergraduate programs that emphasize rigorous V&V understandings, cross-cutting programs *and* faculty will be required. Those that model analytically (mathematicians, physicists, chemists) should be involved in chemistry, biology and physics labs to see, first hand, the limits of the equations that form their metaphysical models. Those that model computationally need to understand the basic

components and fundamental limitations of the underlying computational system that supports the implementation of their algorithms. There are many other benefits from looking outside one's research box. Physical phenomena often have inherent, observable parallelism either due to natural dependencies or physical constraints; this inherent parallelism could be embodied in the mathematical model, implemented in the algorithmic solution, realized in the program design, leveraged by the programming language (HPF, F90 or through MPI, for example) and supported by the underlying communication topology of the computational environment. But this breadth of understanding is exactly what an undergraduate institution should strive to convey.

To address the issue of undergraduate education in V&V, a pedagogical paradigm shift that includes cross-disciplinary exposure to wider issues surrounding the modeling process will be required. A program strong in V&V underpinnings will include appropriate expositions in numerical analysis, error identification, modeling with differential equations, computational algorithms, stochastic analysis, software engineering and probability. While the degree of topic coverage will vary, exposing students to cross-cutting aspects of the modeling process should be common to each discipline.

8. Recommendations

Throughout this document, the underlying assumption has been that V&V is present in every aspect of modeling and simulation, but its cross-disciplinary significance is entirely overlooked in academic programs. The significant risk incurred by the lack of adherence to V&V principles is unbalanced by the minor cost of increasing the presence of V&V in academia.

The cost of increasing the presence of V&V principles to an educational institution can be as economical as simply highlighting current components of V&V that are already rooted in the curriculum. By highlighting V&V aspects and incorporating additional V&V modules throughout the existing curriculum, V&V awareness is raised and the foundation is laid for curriculum support.

Major efforts are required at the academic program level. The leading problems in research areas such as

high-performance computing and bioinformatics will require the support of V&V principles together with interdisciplinary efforts from mathematics, statistics, computer science, and the physical sciences. Academic programs should evolve in order to meet this demand and to supply these areas with researchers capable of applying the breadth of V&V principles. Academic programs must also expand their breadth. The essential components in modeling and simulation reflect the integration of interdisciplinary concepts.

References

- [1] Guide for the verification and validation of computational fluid dynamics simulations, 1998.
- [2] K. Atkinson. *An Introduction to Numerical Analysis, 2nd Edition*, chapter 1. John Wiley, New York, 1989.
- [3] B. M. Ayyub. *Guidelines on Expert-Opinion Elicitation of Probabilities and Consequences for Corps Facilities*. USACE, IWR, 1999.
- [4] O. Balci. Verification, validation, and accreditation of simulation models. In S. Andradattir, K. J. Healy, D. H. Withers, and B. L. Nelson, editors, *Proceedings of the 1997 Winter Simulation Conference*, pages 135–141, 1997.
- [5] J. Blyskal and et. al. *Evolutionary Spiral Process Guidebook*. Software Productivity Consortium, Herndon, VA, 1991.
- [6] CERT. <http://www.cert.org/summaries/CS-2002-01.html>.
- [7] J. Desel. Teaching system modeling, simulation and validation. In *Proceedings of the 2000 Winter Simulation Conference, 2000*.
- [8] J. Gleick. A bug and a crash. sometimes a bug is more than a nuisance. *New York Times Magazine*, Dec 1996.
- [9] C. Harrel, B. Ghosh, and R. Bowden. *Simulation Using Promodel*. McGraw-Hill, 2000.
- [10] B. T. Hill. Lessons learned from the cerro grande (los alamos) fire. *United States General Accounting Office, Testimony before the Committee on Energy and Natural Resources, U.S. Senate*, July 2000.
- [11] S. Hoover and R. Perry. *Simulations: A Problem Solving Approach*. Addison-Wesley, Reading, PA, 1990.
- [12] IEEE. Ieee standards collection. In *IEEE Standards Collection, Software Engineering*. IEEE, New York, NY, 1994.
- [13] D. Isbell and D. Savage. Press release 99-134. NASA, Nov. 1999.
- [14] C. Moler. A tale of two numbers. *SIAM News*, 28(1), Jan. 1995.
- [15] P. G. Neumann. Uss yorktown dead in water after divide by zero. *The Risks Digest*, 19, July 1998.

- [16] W. T. Oberkampf, S. M. DeLand, B. M. Rutherford, K. V. Diegert, and K. F. Alvin. A new methodology for the estimation of total uncertainty in computational simulation. In *AIAA Non-Deterministic Approaches Forum*, 1999.
- [17] Editorial note. *Physical Review Letters*, 89(039901), 2002.
- [18] D. E. Stevenson. A critical look at design, verification, and validation of large scale simulations. *IEEE Manuscript #107243*, To appear in IEEE CSE.
- [19] W. Tucker. *Applied Modeling and Simulation: An Integrated Approach to Development and Operation*, chapter 14, pages 105–125. Space Technology Series. McGraw Hill, 1998.
- [20] P. Wilhide. Press release 00-46. NASA, Mar. 2000.