Constitutive Model Verification and Validation

Len Schwer
`Len@Schwer.net`

Introduction

In computational mechanics the terms Verification and Validation (V&V) are usually applied to discussions concerning large scale software packages (codes) used to provide solutions to practical engineering problems. In this context, one common view of constitutive models is that they represent but one aspect of a code’s solution architecture, that is covered when applying the principles of V&V to the code. An alternate view, presented here, is that constitutive models are stand-alone mechanics software modules, to which the principles of V&V can be applied directly.

Of necessity, the application of V&V principles directly to constitutive models, is limited. The material model validation process needs to be augmented by additional validation that includes the whole of the code in which the module is embedded. Arguments are presented for adopting this two tier, i.e. module and code, approach to constitutive model V&V along with the inherent advantages and limitations.

The following V&V principles (Roache, 1998) are applicable to constitutive models:

Verification - Constitutive models require verification to assure the developer, and especially the user, that the mechanics and numerical algorithms implemented in the module are correct, and that they are consistent with the information provided in the documentation.

Calibration – Constitutive models require the specification of various material constants, or constitutive parameters, for each material. Values for these parameters cannot be derived from first principles, and thus must be obtained by calibration of the model with experimental data. In most cases this must be done indirectly by fitting the predicted material behavior to laboratory data for various load paths.

Code-to-Code Comparisons – The practice of comparing essentially identical constitutive models between and among solid mechanics codes is useful. Agreement between codes does not in any sense constitute a verification of the constitutive model, but any disagreement provides an opportunity to identify and understand code and, hopefully, constitutive model implementation differences. Benchmarking, as this activity is sometimes called, is best performed using material model drivers, rather than simple solid mechanics models.

Validation – Comparisons with structural response experiments is the only practical way to validate a constitutive model. Such validation experiments, i.e. simple solid mechanics problems, can only be analyzed using the computational mechanics code in which the constitutive model is but one module. Thus, the validation of a constitutive model is restricted not only to the specific problem solved, but also to the code in which it is embedded.

Summary of distinctions among terms:

- Verification – code-to-analytical solution comparisons,
- Calibration – material model parameter determination,
- Validation – code-to- experimental data comparisons,
- Code-to-Code Comparisons.

With the exception of Calibration, these are general, but simple, V&V definitions. For constitutive model calibration, a very restricted sense of the term calibration is used, i.e. calibration required to determine the material model parameters.

Verification

Verification of constitutive models *must* be performed by exercising the material models without involving the associated solid mechanics software in which the model is embedded. Most constitutive models are strain-driven, i.e. known increments in strain are input to the constitutive model, and the corresponding stresses are output. Thus by specifying strain increments, and the material characterization parameters, the corresponding stress output can be used for model verification without the need to activate other parts of the associated solid mechanics software. Most commercial solid mechanics software packages provide such strain drivers, which are generally referred to as Material Model Drivers.

Although most constitutive models are strain-driven, many paths of interest for verification are stress paths, e.g. uniaxial stress. Typical Material Model Drivers cannot handle stress-driven paths because the constitutive models are formulated to be strain-driven. The work-around for this dilemma is to use a single cell model with the appropriate stress boundary conditions to exercise the constitutive model through the desired stress path.

The use of a single cell ‘material model driver’ is a practical alternative, but unacceptable for purposes of constitutive model verification. The practical aspect of this is that it is better to perform some *check* on the constitutive model than none. The unacceptable aspect is the realization that the single cell model brings with it many unintended and unwanted aspects of the code’s solution architecture.

Commercial solid mechanics software is designed to be robust. In part, that robustness is achieved by pre-selecting (default) values for the numerous non-material parameters that are included in *all* solutions. Depending on the vendor and the software package, some of these parameters may be available to the user for adjustment while others may not. In any case, any adjustment, or selection, of such parameters is not within the purview of constitutive model verification.

Illustration – Artificial bulk viscosity, in explicit integration software, is essentially a vestige of the hydrocode heritage of most solid mechanics codes. Artificial bulk viscosity is intended to smear shock wave fronts over several cells, and hence to improve the robustness of these codes, for example, for high speed or hypervelocity impact simulations. The viscosity is ‘tuned’ to be active at high strain rates, e.g. at a shock front, and to be negligible at low strain rates. For the majority of computational solid mechanics applications, the bulk viscosity remains negligible, and many users are unaware of it, and certainly unaware of its potential effects. While the most common occurrence of high strain rates is low-to-moderate straining occurring over a very short time duration, e.g. at shock front, there is another material-dependent occurrence. Consider a crushable foam material being compressed in a vehicle impact simulation, i.e. low nominal strain rate. Although the time increments are relatively large, compared to those in a shock wave simulation, the strains in a given time increment can also be large due to the low stiffness of the crushable foam. The likely result is a localized high strain rate, at the crush front, which triggers the artificial bulk viscosity, with the *unexpected* result that the crushable foam material is significantly (and artificially) stiffened due to the *unwanted* viscosity. See Schwer and Whirley (1996) for a more detailed explanation and example.

Illustration – For strain-rate sensitive material models, verification of the rate effects using single cell models of necessity involves the material’s mass to solve the equations-of-motion, and hence the results obtained from the single cell will depend on the cell size. This is particularly obvious when the simulated loading is stress driven, e.g. uniaxial stress. Then the inertial confinement provided by the single cell’s nodal masses can significantly increase the mean stress in the single cell model and in the case of pressure sensitive constitutive models, e.g. frictional materials, affect the material’s strength.

Designing a Constitutive Model Verification Suite

This section provides a list of suggested types of studies for constitutive model verification.

General Constitutive Model Features

Code Checking – The purpose of reviewing the source code of a constitutive model is two fold:

1. To verify, by inspection, that the mechanics and algorithms represented in the material model's documentation are consistent with the code implementation.
2. To scan the code for potential verification weaknesses, i.e. things that can cause it to fail or to behave poorly:
    - Potential division by zero,
    - Differences in nearly identical large numbers not producing zero,
    - Use of ad hoc tolerance limiters (often unit-dependent),
    - Iteration loop limits.

As discussed by Roache (pp 24-25, 1998), reading source code does not imply verification, but should improve user confidence in the documentation and can highlight areas to be investigated with other verification techniques. Consideration may also be given to the application of other well established computer science forms of code verification, see for example Balci (1994) or the evolving web document *VV&A Recommended Practices Guide.*

Prescribed Strain and Stress Paths – Comparing the constitutive model output with analytical solutions for various strain and stress paths is the core of material model verification. Unfortunately, few analytical solutions exist for most material models, and those that do employ fairly simple paths that almost certainly assume proportional straining or loading. Even when analytical solutions exist, they are often couched in mathematics that requires implementation in an algorithm to obtain comparative numerical results. At this point the distinction between verification and code-to-code comparisons becomes unclear.

Unloading and Reloading – Among the strain and stress paths used for verification, it is important to include paths that exercise unloading and reloading. These paths test the constitutive model's ability to return elastically to the same state, and should be continued even further to exercise any form of isotropic or kinematic hardening included in the plasticity formulation of most engineering material models.

Strain Increments – The increments of strain used in the prescribed paths should be varied over a wide range to test for algorithmic weaknesses, e.g. very small increments producing round-off errors that destroy the solution and large increments that cause the algorithm to fail to converge.

Strain Magnitudes –Constitutive models should be tested at a range of strains spanning infinitesimal through finite to large strains to assess their ability to provide accurate solutions as the total strain increases. Both the rate of deformation (stretching) and spin (vorticity) tensors need to be tested in the finite and large strain and rotation regimes.

System of Units – Constitutive models with embedded tolerances or other 'hard wired' numerical parameters may suffer a degradation in performance when a system of units that produces relatively large numerical values for material parameters is tested, e.g. metric systems using grams-centimeters-seconds. Similarly, systems of units that produce very small numerical values should also be tested.

Specific Constitutive Model Features

A constitutive model may be thought of as a collection of sub-models where each sub-model plays a specialized role, e.g. yield surface, rate effects, damage accumulation, etc. This view of a constitutive model is useful in examining the model to determine what components (sub-models) need to be verified. The following is a list of some common components in many engineering constitutive models.

Non-Associative Models – Constitutive models that use non-associated plasticity algorithms need to be examined for their potential to produce unstable material model response, e.g. generation of energy in a closed loading cycle.

Strain-Rate Effects – Constitutive models that include strain-rate effects need to be exercised over a broad range of strain rates to assess model performance and accuracy.

Thermal Effects – Constitutive models that include thermal effects need to be exercised over a broad range of temperatures to assess model performance and accuracy.

Softening Effects – Constitutive models that include strain-softening effects need to be exercised over a broad range of softening responses to assess model performance and accuracy.

Failure– Constitutive models that include failure, e.g. a zero stress material response under applied load, need to be exercised over a range of potential failure scenarios to assess model performance and accuracy.

Design of Experiments and Input-Parameter Sensitivity

Verification simulations should be conducted as a serially-designed suite. The goal of a well-designed verification suite should be to test combinations of input parameters spanning a wide, multi-dimensional parameter space, with the objective of determining the model's sensitivity to changes in those parameters. To facilitate the selection of parameter combinations and ranges, the established techniques of “Design of Experiments” (Myers and Montgomery, 1995) should be considered.

Calibration

Calibration is essential to constitutive models.

Calibration is the adjusting of computational simulation parameters with the goal of obtaining a specified output, e.g. correlation with experimental data. When applied in-the-large to solid mechanics simulations, the number and range of free parameters available to the analyst is essentially infinite. As used here, in the context of material model calibration, the number of parameters is limited to those needed to describe the constitutive model response, and the range should closely reflect the corresponding data for material characterization

It is important to recognize that even a well-designed material characterization test, e.g. a uniaxial tension test, is not a *material point test*, but rather a sophisticated *structural test*. Here the phrase ‘material point’ test is taken to mean the idealized material response as occurs via the application of a material model driver to a constitutive model. The phrase ‘structural test’ recognizes that in a well-designed and executed material characterization test, the experimental results represent material averages over a specimen and not pointwise material response.

Illustration – In a uniaxial stress tension test for metals, the test specimen is a flat piece of metal shaped, as specified by ASTM standards, in a ‘dogbone’ like form. The ends of the specimen are gripped in the testing fixture and extensometers, for determining strains, are placed near the center of the specimen. To provide an estimate of the elastic (Young’s) modulus:

- The machine’s recorded load is divided by the original area of the specimen in the vicinity of the strain gage to produce an average stress,
- The *corresponding* strain is recorded over the length of the extensometer near the center of the specimen,
- The ratio of these two remote, from each other, measured averages provides the elastic modulus.

Assuming the test is well-performed, e.g. specimen alignment, gage placement, attachment and calibration are adequate, the resulting elastic modulus still represents an average material response based on mutually remote measurements and not a pointwise material response. While this illustration of material characterization is trivial, especially for most metals, it is intended to underline a subtle point. The reader needs to keep in mind that especially for non-metals, material characterization tests are more complex, and require much more ‘interpretation,’ than is obvious to most analysts, and even to some experimentalists.

Illustration – For strain-rate sensitive material models, calibration, and as illustrated above verification, of the rate effects using single cell models of necessity involves the material’s mass to solve the equations-of-motion, and hence the results obtained from the single cell will depend on the cell size. This is particularly obvious when the simulated loading is stress driven, e.g. uniaxial stress. Then the inertial confinement provided by the single cell’s nodal masses can significantly increase the mean stress in the single cell model and in the case of pressure sensitive constitutive models, e.g. frictional materials, affect the material’s strength. While a Material Model Driver is required to verify the strain-rate implementation, the use of only a Material Model Drive may not be adequate for calibrating strain-rate effects. If the laboratory material strain-rate characterization technique provides significant inertial confinement of the specimen, then simulations of the laboratory technique may be required for model calibration, with the warning that these simple structure simulation results may be cell sized dependent.

Code-to-Code Comparisons

The practice of comparing code-to-code outputs for a well-defined problem, is sometimes referred to as benchmarking. It is appropriate to remember here the distinctions among terms:

- Benchmarking – code-to-code comparisons
- Verification – code-to-analytical solution comparisons
- Validation – code-to-experimental data comparisons

Because the number and type of analytical solutions for most constitutive models is very limited, e.g. simple proportional loading paths, perhaps the best way to exercise the non-trivial load paths, typical in applications, is via code-to-code comparisons.

Code-to-code comparisons for constitutive models can be especially useful when material model drivers are used, because the code-to-code variation in the solid-mechanics related software of the codes is avoided.

Validation

As discussed above, validation of constitutive models as stand-alone code modules is not possible in a strict sense. The constitutive model to be validated needs to be exercised as part of a larger computational mechanics software package that models a laboratory experiment.

The simplest constitutive model validation experiments are the material characterization tests used to calibrate the material model parameters, e.g. uniaxial tensile stress test. The analyst could construct a simple structural model of the test specimen, i.e. a dogbone tensile coupon in the case of a uniaxial tensile stress test, and replicate the laboratory loading and measurements. However, if the same material characterization test had been used to calibrate the material model parameters, then success, or failure, of the structural model comparison with the calibration data reflects on the solid mechanics code and not the constitutive model, i.e. the verified constitutive model should reproduce the data used in its calibration.

Illustration - Consider a uniaxial stress test where the applied load is incorrectly reported as twice the load applied to the test specimen. The material model's elastic modulus would thus be calibrated to twice the material's true modulus. When the corresponding simple structural model is exercised, the experimentally determined incorrect load will be applied to the structural model and the computed results will 'validate' the incorrect elastic modulus.

What is needed are very simple laboratory experiments, i.e. simple model tests, that exercise the critical features of the constitutive model and are independent of the material characterization tests.

Illustration – Assume that a standard uniaxial tensile stress test has been performed and used to calibrate the material parameters for a constitutive model. A corresponding simple laboratory model test would be internal pressurization of a closed-end tube made of the same material. The pressure history and axial and circumferential strains would be measured. The strain data would be compared directly with the output from a model of the experiment. The data., along with the well-know mechanics relations for stress in a thick walled tube (cylinder), can be used to synthesize the data and generate other material model validation information, e.g. effective stress versus effective strain for a power law hardening material model.

Error Estimation

Discussions of error estimation are not specific to constitutive models, but apply more broadly to verification of all results from numerical simulations. This important topic is discussed by Ainsworth and Oden (2000) and Oberkampf, et al. (2000)

Material Model Drivers

Developing strain-driven material model drivers is relatively straight-forward, and most commercial solid mechanics codes provide the user with this type of driver. Developing stress-driven material model drivers requires considerably more effort. In general, some form of iterative approach is required to select trail-strain increments to drive the constitutive model, and to then test to see if the resulting stresses meet the prescribed stress path.

The main argument against developing stress-path-driven material model drivers is that they replicate a portion of the solution algorithm already implemented in the solid mechanics code.

The arguments for developing stress-driven material model drivers include:

1. Stress paths are the most useful, and most important loading paths, and thus worth verifying and benchmarking, as well as possible. It should again be noted that most material characterization tests are stress-path driven, e.g. uniaxial tensile stress, and these need to be treated with a stress based material model driver.
2. An iterative solution procedure is required, but this should be external to the solid mechanics code, and should be made available to the user for modification and verification unto itself. This also avoids proprietary concerns of commercial software vendors.
3. The effort to develop such a stress-path driver can be shared with other similar constitutive models in the same code, and across codes that use the same or similar material models.
4. A stand-alone iterative solution procedure, especially one provided as open source, is guaranteed not to include any numerical parameters associated with robustness of the associated solid mechanics software. This facilitates the process of code-to-code comparisons of constitutive models.

Stress-driven material model drivers are basically virgin territory in computational solid mechanics because code users use the single cell driver work around. The goal of a general material model driver would to be to allow the user to specify any combined stress/strain path for any material model; reality will limit this goal.

Material Model Documentation

Although documentation may normally not be thought of as part of the verification and validation process, it is essential to an efficient assessment of constitutive models. This is especially true with the extensive reliance on general purpose commercial codes, where most often the source code in not available. Undocumented constitutive models, including those with only descriptions of the input parameters, should be avoided. The task of providing at least an initial verification and validation assessment of a constitutive model is the responsibility of the constitutive model developer. Providing documentation, both theoretical and user manual, is ultimately the responsibility of the code developer (vendor) who chooses to include selected constitutive models as part of their solid mechanics product.

This section provides a list of suggested types of documentation for constitutive models, with the goal of supporting supplemental user verification and validation. The suggested presentation order is intended for new users of the associated constitutive models, experience users would refer to a terse user manual description. The documentation is organized in levels of increasing technical detail and sophistication, so the user may enter and exit the documentation at appropriate interest levels.

Level 1 Overview - to include statements of model applicability, capabilities and limitations.

Level 2 Example Problem – a simple, but practical, problem description that illustrates the application of the material model and the computed results. It is not the intent to provide an echo of a particular code's input/output but rather a problem description and the expected results, i.e. textbook example. Such an example problem should be designed to be a validation problem and is part of the constitutive model's V&V assessment.

Level 3 Determination Input Parameters –descriptions of the material testing required to determine the model parameters with example descriptions and results based on either material model driver examples, or simple model test simulations that illustrate the material testing procedures. The examples in this section can be used as part of the Calibration and Validation assessment of the constitutive model.

Level 4 Tutorial Based Theoretical Description – initially this can be a simple recitation of the model's basic mathematical representation, but should evolve into a stand alone teaching module that the interested user can use to gain insight into the model's development. This section should educate the user, through demonstration, of the statements of model's applicability, capabilities and limitations given in the Overview Level.

Layer 5 Numerical Implementation – a description of how the Theoretical Description was translated into numerical algorithms. When possible, this description might best be augmented as a narrative interspersed with a listing of the constitutive model coding.

References

Ainsworth, M., and JT Oden, *A Posteriori Error Estimation in Finite Element Analysis*. Wiley – Interscience, NY, ISBN 0-471-29411-X, 2000.

Balci, O, "Validation, Verification, and Testing Techniques Throughout the Life Cycle of a Simulation Study," *Annals of Operations Research*, Volume 53, pp.121-173, 1994.

Myers, RH, and DC Montgomery, *Response Surface Methodology, Process and Product Optimization Using Designed Experiments*, John Wiley & Sons, New York, 1995.

Oberkampf, WL, SM DeLand, BM Rutherford, KV Diegert and KF Alvin, *Estimation of Total Uncertainty in Modeling and Simulation*, Sandia National Laboratories, SAND2000-0824, April 2000.

Roache, PJ, *Verification and Validation in Computational Science and Engineering*, Hermosa Publishers, ISBN 0-913478-08-3, Albuquerque, NM, 1998.

Schwer, L.E. and R.G. Whirley, "Lessons Learned in Modeling a Moving Deformable Barrier (MDB) Impacting a Rigid Wall," *International Journal of Crashworthiness*, Volume 1, Number 1, pp.73-92, 1996.

VV&A Recommended Practices Guide, (`http://www.msiac.dmso.mil/vva/`) Modeling and Simulation Information Analysis Center (MSIAC) a Department of Defense Information Analysis Center, sponsored by the Defense Technical Information Center and the Defense Modeling and Simulation Office.

Acknowledgements

Comments and suggestions by William Scherzinger of Sandia National Laboratories, Thomas Pucik of Pucik Consulting Services, Hans Mair of the Institute for Defense Analyses, Dale Pace of Johns Hopkins University Applied Physics Laboratory, Paul Senseny of Factory Mutual Research, John Cafeo of General Motors, Tinsley Oden of University of Texas at Austin, and Patrick Roache on the early drafts of this manuscript are much appreciated.