The Validation of Computer-based Models in Engineering: Some Lessons from Computing Science

D. J. Murray-Smith

Questions of the quality of computer-based models and the formal processes of model testing, involving internal verification and external validation, are usually given only passing attention in engineering reports and in technical publications. However, such models frequently provide a basis for analysis methods, design calculations or real-time decision-making in complex engineering systems. This paper reviews techniques used for external validation of computer-based models and contrasts the somewhat casual approach which is usually adopted in this field with the more formal approaches to software testing and documentation recommended for large software projects. Both activities require intimate knowledge of the intended application, a systematic approach and considerable expertise and ingenuity in the design of tests. It is concluded that engineering degree courses dealing with modelling techniques and computer simulation should put more emphasis on model limitations, testing and validation.

Keywords: computer-based models, simulation, model limitations, software testing, external validation, software engineering tools.

1 Introduction

Mathematical modelling is central to many aspects of engineering design. This is particularly true in the case of control systems engineering where models of the system to be controlled ("plant" models) play an important part in design. In some forms of control algorithm dynamic models are actually explicitly incorporated within the controller.

Plant models used in design or in the implementation of control systems are often linear in form because most control system design methods are based upon linear theory. On the other hand, evaluation of the overall system performance for a system with a controller designed on a linear basis usually necessitates use of a nonlinear description of the plant which incorporates some features which, although neglected at the initial design stage, have to be taken into account in performance evaluation studies prior to commissioning of the system.

Although it has for long been accepted that an essential part of the modelling and simulation process involves establishing the credibility of a simulation model in the context of the intended application, there is much evidence that, in practice, this aspect of modelling is often treated in a superficial way [1]. Apart from some specific safety-critical applications, little attention appears to be given to model testing and to establishing the quality of models in terms of their useful range and limits of accuracy.

The lack of attention given to external validation of models of engineering systems can lead to expensive redesign at late stages in the development cycle and there are many examples which can be used to illustrate this. Modelling errors are especially important in the design of high-performance automatic control systems where model uncertainties can make it impossible for a control system to meet given performance specifications. It is now becoming accepted that in some application areas dependence upon linear perturbation models for control system design may not be sufficient for high-performance applications [2]. One good example of this is in helicopter flight control systems design where it is now recognised that the success of optimal control and other synthesis methods has been limited by the range and accuracy of available models of the vehicle [3, 4].

Developments in the theory of modelling and simulation [5] provide a useful methodological framework within which to consider issues of verification and validation of simulation models. The concept of an *experimental frame*, which is separate from the underlying system model and the simulation program, is central to present-day theory of modelling and simulation methods.

An experimental frame can be divided into *generator, acceptor* and *transducer* elements. The *generator* is used to stimulate the system and the model with identical input sequences or trajectories (e.g. steady state values, steps, ramps, periodic signals etc.). The *acceptor* is the element through which the user can specify the conditions which are of interest (e.g. steady states, transient behaviour) and limit the observation of behaviour to the specified conditions. The *transducer* element of the experimental frame post-processes the output time histories and extracts the measures of interest.

2 Methods for external validation of computer-based models

External validation of computer-based models involves testing the underlying mathematical model to ensure that its behaviour is consistent with that of the real system that it represents. It can also involve establishing the operating range over which the model is appropriate for the intended application. This process of external validation has to be distinguished carefully from the process of *internal verification* which involves testing the computer simulation program to establish that it is consistent with the mathematical model in terms of its structure and also that it is algorithmically correct in the sense that simulation solutions accurately represent model solutions.

External validation presents a much greater challenge than internal verification to those involved in computer-based modelling and simulation of engineering systems. The approaches to external validation that are most commonly used are based upon comparisons of response data from the model and from the real system, together with appropriate model performance measures (the transducer element within the experimental frame). For example, many different deterministic measures of model quality have been proposed [1] and statistical techniques [6] based on the fitting of auto-regressive moving-average models or stochastic models to time series of the model and real system variables can also be used. If the two models are the same the two time series are equivalent in some respects. Techniques of step-wise regression have also been applied successfully to model structure assessment [7].

System identification and parameter estimation techniques [8, 9] provide an approach to validation that involves comparisons between system and model which are somewhat less direct than in the methods outlined above. Comparisons of parameter estimates for linear empirical models obtained from experiments on the real system with equivalent quantities derived by linearisation of a nonlinear theoretical model may allow conclusions to be reached about the overall validity of the theoretical model and possible sources of error. The concept of *identifiability* [10, 11] is itself of great significance in the establishment of experimental frames for external validation and the interpretation of estimates of unknown parameters. Other tools of potential value for external validation include parameter sensitivity analysis [12, 13], inverse modelling [1, 14] and model distortion methods [15].

Whatever the chosen methodology for external validation, experimental design is of vital importance since the information content of the response data is central to the validity of the conclusions reached regarding the suitability or otherwise of a particular model for a given application. If the frequency or amplitude ranges chosen for the generator are inappropriate the conclusions reached will be of little value. At best the model will be restricted to the range of conditions over which it has been tested.

Documentation of the complete model development process, including all testing, is very important. A complete record should be kept of the aims and objectives of the work, the purpose of the model, the detailed specifications of the model, all assumptions, simplifications and approximations used, tests applied for external validation, experimental records obtained for validation testing and the associated analysis techniques and results. The rationale used in the decision to accept or reject a given model must also appear in the documentation.

In most engineering projects which depend on computer-based models in the design, implementation or application phases, additional evidence of the suitability or otherwise of the model will continue to be accumulated throughout the whole duration of the project. Documentation must therefore be maintained throughout the application phase. Within design organisations it may well be helpful to retain model documentation beyond the lifetime of the real system as this may provide generic information which may be useful for subsequent design projects involving other systems of a similar kind.

3 The relevance of software engineering testing principles

In the early days of computer programming, testing of software was viewed as "debugging" and was carried out by the programmers themselves as a post-development activity. By the 1980s the term "software engineering" was being used to describe the software development process and software testing began to be recognised as a separate activity to be performed by independent testers using appropriate testing tools.

Many definitions of testing are available. One which is particularly appropriate is "verifying that a system satisfies its specified requirements or identifying differences between expected and actual results" [16]. This definition gives emphasis to the fact that during testing one needs to be able to anticipate what is *supposed* to happen and to compare what actually does happen with that prediction.

It is now recognised that in software development projects testing is more than just a phase of work which occurs towards the end of the development cycle [16, 17]. The testing process starts at the stage at which requirements are defined and occurs again at every subsequent stage of the development cycle through design and implementation to operation and maintenance. Clearly the cost of software errors can be minimised if the errors are detected at the stage of development in which they are introduced. In general it is vitally important to prevent the migration of errors from one phase of software development to a later phase.

Most testing involves a bottom-up process in which low-level modules are tested first, with emphasis placed initially at the unit or module level. Higher-level testing, involving integration testing and complete system testing, is carried out at a later stage in the development cycle. In general this makes it easier to establish the cause of any failure.

Whatever approach to software testing is adopted every new version of a unit or product should be retested after modification through a process known as *regression testing*. This involves re-execution of some or all of the tests carried out during the initial (or *progressive*) testing process. Regressive testing puts special emphasis on the need for good documentation of software testing procedures as part of the complete documentation of the software development process.

Testing may be used to show that errors are present but never to prove that they are absent. Effective testing removes errors but in complex applications it may be difficult to know how much testing is appropriate and some form of risk assessment and management may be called for in establishing what is needed. Critical software (as defined by the IEEE/ANSI standards [16]) is software which could have an impact on safety or could cause large financial or social losses if it failed.

4 Discussion

Even a superficial review of recent publications on software testing suggests that the similarities between this activity and the internal verification and external validation of computer-based models are significant. Almost every statement written above about software testing could also be presented in the context of good practice for model development and becomes especially clear when viewed from the perspective of the experimental frame. For example, testing at the module or unit level is just as important in computer-based modelling and simulation as it is in software engineering. As with software, complete testing of a model is impossible because the domain of possible inputs is too large to test exhaustively and realistic test planning involves selecting a small number of test cases which are designed to detect errors. Careful consideration has to be given to the generator, acceptor and transducer elements within the experimental frame. The amount of testing to be carried out and the extent to which exhaustive regressive testing is applied depends on the consequences of possible model errors or failure of the model and is therefore a matter for risk assessment. In safety-critical applications it is clear that model testing procedures and model documentation processes are already much more rigorous than in other areas.

While parallels exist between software testing and model testing it is clear that computer professionals are generally more aware of testing methodologies than are most engineers who use modelling and computer simulation tools. Although lip-service may be paid frequently to issues of model quality in applications in which models have a central role in design, relatively little attention appears to be given, in practice, to systematic model testing and to model documentation.

In spite of the enormous advances of software engineering in the past thirty years the testing process is still relatively immature in most organisations and testing still does not receive the attention which it should within the academic world. Nevertheless, the situation in terms of systematic testing of software is very much more satisfactory than is the case for systematic and exhaustive testing within the model development process.

The apparent lack of awareness of engineers about the processes of testing and documentation of models points to underlying problems in their education. It appears that most courses which deal in some way with modelling and simulation issues at university level put most emphasis on model formulation and on numerical methods. More difficult questions about model accuracy and fitness for purpose are very often neglected or treated in a superficial way.

Some aspects of the problem could be handled more easily if there was wider understanding of the tools used by software engineers within the software development process. For example version control techniques are routinely used in software development but are seldom applied in a rigorous fashion for simulation model development, maintaining the model throughout its life cycle [18] and documentation. The tools are readily available and could be applied in modelling just as effectively as in software engineering.

The tasks involved in developing a simulation model extend far beyond the technical processes of constructing a computer-based description of a set of mathematical equations and the conduct of simulation experiments. The processes involved in investigation of the accuracy and limitations of a model may include analysis of linearised descriptions derived from a more general nonlinear model, storage, retrieval and quantitative comparison of simulation and experimental results for a wide range of test conditions, system identification and parameter estimation, sensitivity analysis, experimental design, post-processing, visualisation and documentation (not only of the model itself but all the associated external validation experiments). These wide-ranging requirements strongly suggest that there are potential benefits to be obtained from the use of a properly integrated set of software tools covering continuous system simulation, optimisation routines, database software for experimental and simulation model response records, and facilities for visualisation [19]. Such an integrated set of software tools should be made available within a well defined and properly managed software engineering environment.

There is already awareness of these methods in safety-critical areas of engineering such as aircraft flight control system desgn, flight simulator development and in the nuclear industry, where they are already being applied successfully. The systematic approach to model development and maintenance could and should be extended to other areas of engineering application where they could offer significant benefits in terms of reduced development time, reduced risk and potentially better performance. Achieving these benefits is largely a matter of education of those who are likely to be engaged in the development and use of computer-based models and re-education of many who are in this field already but are unaware of the potential benefits of using a properly integrated and controlled software environment and more appropriate techniques for model validation.

5 Conclusions

It is accepted that the quality of a software system is primarily determined by the design specification, by the quality and effectiveness of the development process and by the commitment of all concerned in the project to excellence. Systematic testing of software is an essential element of that development process. Those engaged in software development activities are well served by excellent computing environments which can assist greatly in the management of complex development processes. Many of the same issues arise in the development and testing of models used within engineering applications. The final quality and suitability of a model depends upon the appropriateness of the specification (in the context of the intended application), the nature of the development process used and the skills of those involved, especially in terms of testing.

The qualities required by computing professionals involved in software development and testing and by engineers engaged in the development and testing of models are very similar. It is believed that engineering degree programmes should give increased emphasis to the modelling process, including external validation principles. Documentation of models should be emphasised and the dangerous consequences of inadequate documentation should be stressed during the training of students. Development tools and principles widely used for software development projects should be adapted for engineering applications which involve the development and use of computer-based models.

References

[1] Murray-Smith, D. J.: *Methods for the external validation of continuous system simulation models: a review.* Mathematical and Computer Modelling of Dynamical Systems, Vol. 4, 1998, pp. 5–31

[2] Murray-Smith, D. J.: *Issues of model accuracy and external validation for control systems design*. Acta Polytechnica, Vol. 40, 2000, pp. 8–13

[3] Tischler, M. B.: *System identification requirements for high-bandwidth rotorcraft flight control systems*. J. Guidance, Control and Dynamics, Vol. 13, 1990, pp. 835–841

[4] Tischler, M. B.: *System identification methods for aircraft flight control development and validation*. In M. B. Tischler (Ed.), "Advances in Aircraft Flight Control", Taylor and Francis, London, 1996, pp. 35–69

[5] Ziegler, B. P., Praehofer, H., Tag Gon Kim: *Theory of Modeling and Simulation* (2^{nd} *Edition*). Academic Press, San Diego, 2000

[6] Kleijnan, J. P. C.: *Verification and validation of simulation models*. European J. Operational Research, Vol. 82, 1995, pp. 145–162

[7] Draper, N. R., Smith, H.: *Applied Regression Analysis*. Wiley, New York, 1966

[8] Goodwin, G. C., Payne, R. L.: *Dynamic System Identification: Experiment and Design*. Academic Press, New York, 1977

[9] Beck, J. V., Arnold, K. J.: *Parameter Estimation in Engineering and Science*. Wiley, New York, 1977

[10] Bellman, R, Astrom, K. J.: *On structural identifiability*. Math. Biosci., Vol. 7, 1970, pp. 329–339

[11] Murray-Smith, D. J., Bradley, R., Leith, D.: *Identifiability analysis and experimental design for simulation model validation*. In Maceri, F. and Iazeolla, G. (Eds.) EUROSIM'92 Simulation Congress (Proceedings of the 1992 EUROSIM Conference, Capri, 28 Sept.–4 Oct. 1992), North Holland, Amsterdam, 1992, pp.15–20

[12] Tomovic, R.: *Sensitivity Analysis of Dynamic Systems*. MIT Press, 1962

[13] Eslami, M.: *Theory of Sensitivity in Dynamic Systems*. Springer-Verlag, Berlin, 1994

[14] Bradley, R., Padfield, G. D., Murray-Smith, D. J., Thomson, D. G.: *Validation of helicopter mathematical models*. Trans. Inst. Measurement and Control, Vol. 12, 1990, pp. 186–196

[15] Cameron, R. G.: *Model validation by the distortion method: linear state space systems*. IEE Proc.-D., Vol. 139, 1992, pp. 296–300

[16] Kit, E.: *Software Testing in the Real World*. Addison-Wesley, Harlow, England, 1995

[17] Kaner, C., Falk, J., Hung Quoc Nguyen: *Testing Computer Software* (2^{nd} *Edition*). Wiley, New York, 1999

[18] Sisle, M. E.: *Validation throughout the simulation life cycle*. In Proc. 1985 Summer Computer Simulation Conference, July 22–24, 1985, Chicago, Illinois, The Society for Computer Simulation, San Diego, 1985, p. 168

[19] Murray-Smith, D. J.: *Enhanced environments for the development and validation of dynamic system models*. Mathematics and Computer in Simulation, Vol. 39, 1995, pp. 459–464

Prof. D. J. Murray-Smith
e-mail: djms@elec.gla.ac.uk

Centre for Systems and Control and
Department of Electronics and Electrical Engineering
University of Glasgow
Glasgow G12 8QQ
Scotland, UK