On numerical solver selection and related uncertainty terminology

Petra Claeys, Ann van Griensven, Lorenzo Benedetti, Bernard De Baets and Peter A. Vanrolleghem

ABSTRACT

Mathematical models provide insight into numerous biological, physical and chemical systems. They can be used in process design, optimisation, control and decision support, as acknowledged in many different fields of scientific research. Mathematical models do not always yield reliable results and uncertainty should be taken into account. At present, it is possible to identify some factors contributing to uncertainty, and the awareness of the necessity of uncertainty assessment is rising. In the fields of Environmental Modelling and Computational Fluid Dynamics, for instance, terminology related to uncertainty exists and is generally accepted. However, the uncertainty due to the choice of the numerical solver and its settings used to compute the solution of the models did not receive much attention in the past. A motivating example on the existence and effect of numerical uncertainty is provided and clearly shows that we can no longer ignore it. This paper introduces a new terminology to support communication about uncertainty caused by numerical solvers, so that scientists become perceptive to it.

Key words | mathematical models, numerical solver, terminology, uncertainty, uncertainty matrix

Petra Claeys (corresponding author)
Lorenzo Benedetti
BIOMATH and KERMIT,
Department of Applied Mathematics,
Biometrics and Process Control,
Ghent University,
Coupure links 653, B-9000 Ghent,
Belgium
E-mail: *pclaeys@biomath.ugent.be*

Ann van Griensven
HIKM, Hydroinformatics and Knowledge Management,
UNESCO-IHE Institute for Water Education,
Westvest 7, 2611 AX Delft,
The Netherlands

Bernard De Baets
KERMIT, Department of Applied Mathematics,
Biometrics and Process Control,
Ghent University,
Coupure links 653, B-9000 Ghent,
Belgium

Peter A. Vanrolleghem
modelEAU, Département de génie civil,
Pavillon Pouliot, Université Laval, Québec QC,
Canada G1K 7P4

INTRODUCTION

The issue of uncertainty was already emphasised by Einstein when he noted that: "As far as the laws of mathematics refer to reality, they are not certain, and as far as they are certain, they do not refer to reality". The mathematical or numerical approximations of solutions of differential equations, our limited computational capacity and our essential lack of full understanding of the laws of physics and biology all influence the accuracy of complex environmental model simulations. The ability of a numerical method to solve a problem with high accuracy can certainly be demonstrated but, as stated by Einstein, we should always discern between reality and model simulations (Freitas 2002).

Computer simulations have become valuable to gain insight into a plethora of systems such as natural systems in physics, chemistry and biology; human systems in economics, psychology and social science, and engineering of new technology. They could also be a convenient tool in decision support, but caution is advisable.

The degree of confidence that a decision-maker or scientist has in the possible outcomes of a computer simulation is formalised among researchers using the term "uncertainty". For reliable decision support it should be possible to identify the factors contributing to uncertainty and an assessment of uncertainty should be performed.

In the field of Environmental Modelling, Refsgaard *et al.* (2007) introduced a terminology and classification of uncertainty. A framework for the modelling process is presented and the role of uncertainty at different stages of

doi: 10.2166/hydro.2009.072

the modelling process is discussed. In their work little attention goes to uncertainty caused by the choice of the numerical solver and its settings. In the remainder of this work, the "numerical solver" is used to indicate the numerical methods and their implementations used to compute the solutions of the differential equations.

The Numerical Unit Spread Assessment Pedigree (NUSAP) glossary (van der Sluijs *et al.* 2003) contains some information on the source of uncertainty due to the numerical solver. In the fields of Computational Fluid Dynamics (Freitas 2002) and Computational Engineering and Physics (Oberkampf *et al.* 2004) concern about "solving the equations correctly" has been of interest for some time and is formalised as "verification". The process of verification can be divided into two parts: code verification and solution verification. Code verification focuses on how correctly the numerical solvers are implemented and on Software Quality Assurance. Solution verification deals with the quantitative estimation of the numerical accuracy of a solution of a differential equation when solved with a particular numerical solver.

Although many computational results depend on the uncertainty related to the numerical solver, this source of uncertainty has not received much attention in the past. Because of the importance to explicitly consider uncertainties related to the numerical solver, we propose a new terminology to formalise it so that the level of this uncertainty shifts from total ignorance to at least recognised ignorance.

UNCERTAINTY TERMINOLOGY IN THE ENVIRONMENTAL MODELLING PROCESS

In the field of Environmental Modelling, the distinction between different types of uncertainty and the use of a correct terminology for them has been of interest and has resulted in a generally accepted terminology (Refsgaard *et al.* 2007). In this terminology three aspects of uncertainty are considered: the level, the nature and the source of uncertainty.

The level of uncertainty characterises the degree of knowledge. It ranges from total ignorance to deterministic understanding, the latter being impossible to achieve. In between these extremes, different levels of uncertainty can occur. It is useful to distinguish between bounded uncertainty and unbounded uncertainty. In the case of bounded uncertainty all possible outcomes are deemed known, whereas in the case of unbounded uncertainty, some or all possible outcomes are deemed unknown. A lower level of distinction is made between the knowledge of all probabilities of all outcomes, some probabilities or none at all. Statistical uncertainty occurs in the case of bounded uncertainty and knowledge of all probabilities. Qualitative uncertainty is a term used for bounded or unbounded uncertainty, while some outcomes and some probabilities are known (Walker *et al.* 2003).

For the nature of uncertainty two extremes exist: epistemic uncertainty and stochastic uncertainty. Epistemic uncertainty is caused by the imperfection of knowledge, the limited accuracy of measurements and the limitations of state-of-the-art technologies. This nature of uncertainty is reducible by more research and development. Stochastic uncertainty is due to the chaotic, unpredictable nature of natural processes, and due to human behavior together with social, economic and cultural dynamics. Additional research cannot reduce stochastic uncertainty. Stochastic uncertainties can also be the ones that the modeller or model user decides not to analyse and reduce, rather than being fundamentally irreducible. A clear distinction between these two categories is not always easy, because it is difficult to determine what is reducible by research and what is an inherent property of the phenomena under consideration (Walker *et al.* 2003).

Refsgaard *et al.* (2007) define five sources of uncertainty: context and framing, input uncertainty, model structure uncertainty, parameter uncertainty and model technical uncertainty. The uncertainty matrix provides an overview of the various facets of uncertainty in a modelling process (see Figure 1). The vertical axis of this matrix identifies the source of uncertainty, while the horizontal axis considers the level and nature of uncertainty (Refsgaard *et al.* 2007); these two categories are not mutually exclusive. Model output uncertainty is an accumulated uncertainty caused by the uncertainties of context, model and inputs

Source of uncertainty		Level of uncertainty				Nature of uncertainty	
		Statistical uncertainty	Scenario uncertainty	Qualitative uncertainty	Recognized ignorance	Epistemic uncertainty	Stochastic uncertainty
Context	Natural, technological, economic, social, political						
Inputs	System data						
	Driving forces						
Model	Model structure						
	Technical						
	Parameters						
Model output							

Figure 1 | The uncertainty matrix, adapted from Refsgaard *et al.* (2007). More insight into the system under consideration is needed to specify the type and nature of the uncertainties.

(see Figure 1). Several other methodologies for uncertainty assessment exist and comprehensive descriptions are available (van der Sluijs *et al.* 2003).

UNCERTAINTY IN COMPUTATIONAL ENGINEERING AND PHYSICS

In the field of Computational Engineering, "solving the equations correctly" has been of concern to a certain degree. An effort to quantify the numerical error of the computations is made and tests to ensure the correctness of the implementation of the numerical solver exist (verification). Whether the numerical solver is suitable for the problem under consideration is not discussed, however.

Validation analyses the relation between the computations which result from the computerised model and the real world. The real world is represented as experimental measurements from purposefully designed validation experiments (Oberkampf *et al.* 2004).

Verification is the assessment of the accuracy by comparing the solution of a computerised model with known solutions. The most comprehensive and rigorous method to verify the code of a numerical solver is the association of the method of manufactured solutions with the order of accuracy criterion (Salari & Knupp 2000). The order-of-accuracy criterion is fulfilled if the measured order of accuracy of the numerical solver is equal to the theoretical order of accuracy of the numerical solver. For example, for a theoretical order of accuracy of 2 the root mean square error (RMS error) between the analytical and numerical solution should decrease by a factor of 4 when the grid cell size or step size is halved. For this criterion the analytical solution of a test problem is required.

The Method of Manufactured Solutions (MMS) allows us to manufacture such solutions without knowledge of the exact analytical solution (Salari & Knupp 2000). But this method can only be applied to a narrow range of physical models for which the computation of the numerical solution is easy and accurate. The models and manufactured solutions (Salari & Knupp 2000) must follow a multitude of guidelines, and this method also requires symbolic manipulation operations.

Verification is also done by the use of highly accurate solutions, or benchmark ODE solutions, as known solutions. The accuracy of these benchmark ODE solutions becomes important if the exact solution is unknown and one moves away from analytical solutions (Oberkampf *et al.* 2004). To ascertain that simulations are highly accurate without any analytical solution, some rules of thumb were developed, but these rules do not guarantee that the simulations are satisfactory.

Oberkampf *et al.* (2004) explain that the Society of Computer Simulations (SCS) distinguishes two types of models: a conceptual model and a computerised model. The conceptual model is a definition of the mathematical representation of the physical system or process of interest.

This mathematical representation is realised by means of mathematical equations and mathematical modelling data. Computerisation of the conceptual model produces the computerised model.

THE SCALE OF NUMERICAL SOLVER UNCERTAINTY VERSUS THAT OF PARAMETER UNCERTAINTY

The model used by Seppelt & Richter (2005) is an undeniable illustration of the importance of numerical solver uncertainty. This model is a modification of the widely known Lotka–Volterra model that describes predator(y)–prey(x) interactions. It produces completely different (and wrong) results when a different numerical solver is chosen. The equations of the model are shown below, while the parameters are explained in Table 1; for these parameters the system has a stable limit cycle:

$$\frac{dx}{dt} = rx\left(1 - \frac{x}{K}\right) - \alpha\left(\frac{x^2}{x^2+L}\right)y \quad \frac{dy}{dt} = \gamma\alpha\left(\frac{x^2}{x^2+L}\right)y - \mu y \tag{1}$$

Evaluation of numerical solver uncertainty of the Lotka–Volterra model with different solvers

We have simulated the solution of this model within Tornado (Claeys *et al.* 2006), using 29 different numerical solvers that Tornado supports. Twenty-two solvers did not fail (i.e. the computation did not stop before the simulated time was reached). These solvers are: Adams–Bashforth 2 (Hairer *et al.* 1993), Adams–Bashforth 3 (Hairer *et al.* 1993), Adams–Bashforth 4 (Hairer *et al.* 1993), Rosenbrock (Shampine 1982), RK2a (Abramowitz & Stegun 1965), Midpoint (Kloeden & Platen 1995), Euler (Flowers 2000), RK2b (Butcher 2003), Runge–Kutta 4 (Kloeden & Platen 1995), Runge–Kutta–Fehlberg (Press *et al.* 1992), four variations of LSODE (Hindmarsh 1983), four variations of CVODE in combination with two types of Krylov solvers (Hindmarsh *et al.* 2005), DASRT (Hairer *et al.* 1993), DASSL (Brenan *et al.* 1989) and LSODA (Petzold 1983*b*). The seven solvers that failed, had unusual or nonstiff settings were CVODE/Adams/Functional, CVODE/BDF/Functional, CVODE/Adams/Newton/Band, CVODE/Adams/Newton with two types of Krylov solvers, LSODE/Adams/Newton/Diagonal and LSODE/Adams/Functional. A detailed description of the mathematical methods used by these solvers is beyond the scope of this paper. An overview of the use of these numerical solvers for non-expert users is presented in the work of Claeys *et al.* (2007). The accuracy of these solvers was set to a value of 10^{-5}.

As in the work of Seppelt & Richter (2005) the characteristics of the resulting solution trajectories differed significantly. In particular, numerical solvers designed to solve stiff problems produced completely the wrong results. Correct results were computed by 11 solvers only (<50%), namely Euler, Adams–Bashforth 2, Adams–Bashforth 3, Adams–Bashforth 4, Runge–Kutta 4, RK2a, RK2b, Midpoint, Runge–Kutta–Fehlberg, LSODA and Rosenbrock. For comparison, a Monte Carlo experiment was conducted, using a numerical solver that was able to compute the correct solution (AB4). Five hundred sets of model parameters were selected from uniform probability distribution functions, and varied approximately 20% around the mean value (Benedetti *et al.* 2008). The mean values of the parameter values are given in Table 1.

Table 1 | Overview of the model parameters of the modified Lotka–Volterra model used by Seppelt & Richter (2005)

Symbol	Function	Unit	Value
r	Prey growth rate	[time^{-1}]	0.06
K	Prey capacity	[no. of prey]	1,000
a	Predation rate	[time^{-1}]	5
L	Limitation of predation success	[no. of prey^2]	50
γ	Efficiency coefficient of predation	[-]	0.2
μ	Mortality of predator population	[time^{-1}]	0.9
x_0	Initial prey population	[no. of prey]	10
y_0	Initial predator population	[no. of pred.]	0.02
T	Simulated time	[time]	2,000

Comparison of uncertainty due to solver selection and model parameters

Some simple characteristics of the collection of solution trajectories of the two experiments are shown in Figure 2. From this figure it is clear that the uncertainty caused by the choice of the numerical solver, on the left, is much larger

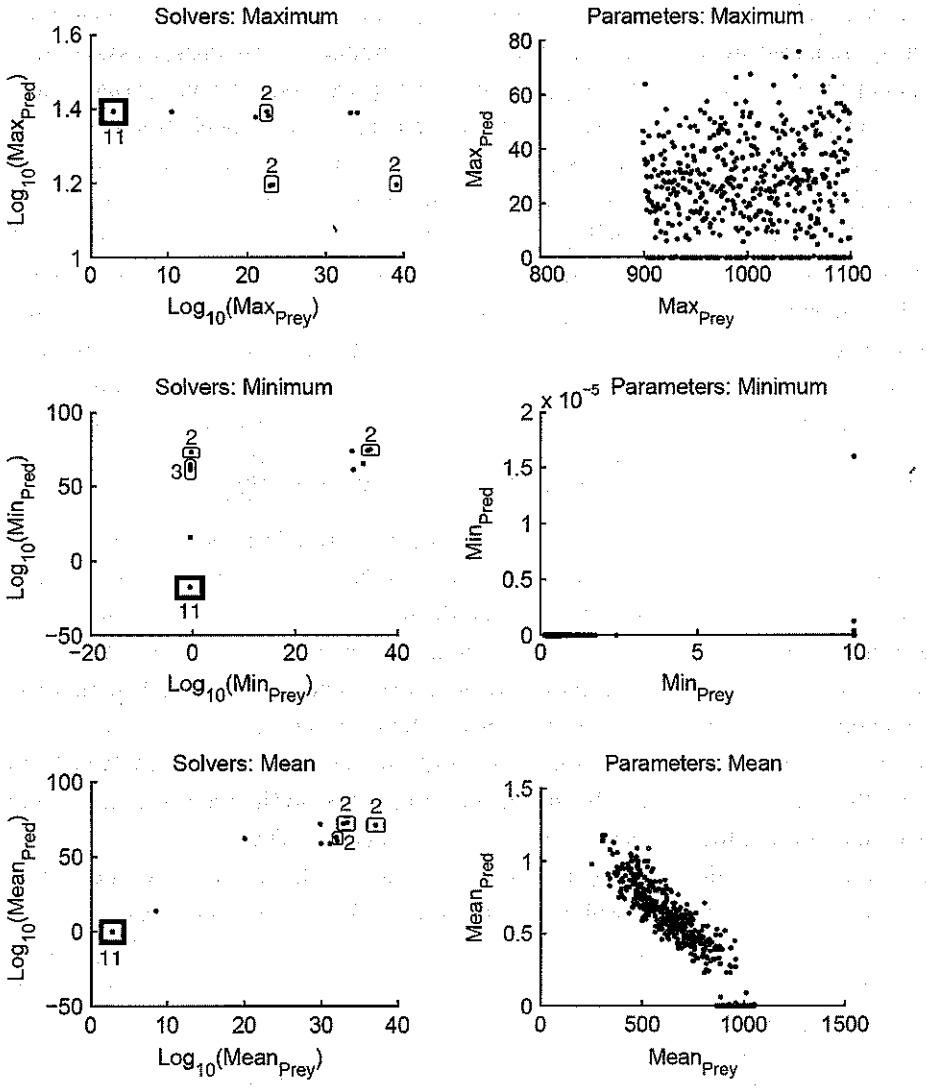

Figure 2 | Overview of the characteristics for the two experiments (solvers on the left and model parameters on the right). Max_{Pred} and Max_{Prey} are the maximum values of the two outputs; Min_{Pred} and Min_{Prey} are the minimum values of the two outputs, and $Mean_{Pred}$ together with $Mean_{Prey}$ are the mean values of the two outputs. The rectangle in bold contains the 11 solvers that computed the correct solutions: $\log_{10}(Min_{Prey}) = -0.368$, $\log_{10}(Min_{Pred}) = -17.45$, $\log_{10}(Max_{Prey}) = 3$, $\log_{10}(Max_{Pred}) = 1.392$, $\log_{10}(Mean_{Prey}) = 2.833$ and $\log_{10}(Mean_{Pred}) = -0.285$.

than the uncertainty caused by the variation of the model parameters, on the right (note that the axes scales differ). This example illustrates that numerical uncertainty can be much larger than parameter uncertainty. Detection of the correct solution trajectories in this case can be done by comparing solution trajectories of several solvers. However, in the case that the correct solution is unknown, many correct solutions must be present so that they become distinguishable from the incorrect ones, which are typically diverging from the correct solution, whereas the correct solutions tend to cluster.

EXTENDING UNCERTAINTY TERMINOLOGY AND METHODOLOGY FOR THE ENVIRONMENTAL MODELLING PROCESS

A detailed description of the newly defined sources of uncertainty, and an explanation of some of the existing methodologies to assess these new sources, follows.

Overview of the extensions

In order to formalise the uncertainties related to the numerical solver that was used to simulate the model,

new sources of uncertainty need to be distinguished. Model technical uncertainty (see Figure 1) can be replaced by a hierarchical structure of newly defined sources of uncertainty (see Figure 3). Making the relation between the newly defined sources hierarchical enables us to define sources of uncertainty that are composed of descendant sources of uncertainty. In other words: some uncertainties actually are accumulated uncertainties caused by their descendants. If, for a source of uncertainty, the level or nature can be deduced in a consistent manner, it is designated by a cross in Figure 3. The remaining part of the matrix can only be completed when all specifications of the model under consideration are known to the user.

Model structure uncertainty

Model structure uncertainty is not a new source of uncertainty. It is explained once more to demonstrate the differences between this source of uncertainty and the newly defined ones. Model structure uncertainty is only related to the mathematical equations that are chosen to describe the system or process of interest (Oberkampf *et al.* 2004). It is a conceptual uncertainty, generated during the modelling process, and caused by incomplete understanding and simplified descriptions of the processes modelled, compared to reality (Refsgaard *et al.* 2007).

Extended peer review (e.g. by stakeholders) can contribute to the reduction of this source of uncertainty. It allows the use of additional knowledge from non-scientific sources to be involved in the quality assurance of the modelling process (Refsgaard *et al.* 2007). In the work of van Griensven & Meixner (2004), model structure uncertainty is considered as the uncertainty that is not caused by parameter uncertainty. Moreover, it is assessed using a split sample approach, which uses half of the dataset for calibration and the other half for evaluation of the model. For more details we refer to van Griensven & Meixner (2004). Several other validation activities can be used to assess model structure uncertainty (Oberkampf *et al.* 2004).

Model computerisation uncertainty

Model computerisation uncertainty is a term that is proposed to cover the collection of uncertainties induced by computerising the model. A computerised model resides on a certain machine, and is an implementation of the conceptual model in a computer programming language or modelling language. In addition, the computerised model comprises a numerical solver; more specifically, it links to an existing implementation of it or it includes a hard-coded implementation of a numerical solver.

The nature of this accumulated source of uncertainty is mainly epistemic, and the limitations of technology are the main contributors to this uncertainty. At this moment no automatic tools exist that allow assessment of this source of uncertainty.

Source of model uncertainty	Level of model uncertainty				Nature of uncertainty	
	Statistical uncertainty	Scenario uncertainty	Qualitative uncertainty	Recognized ignorance	Epistemic uncertainty	Stochastic uncertainty
-Model structure					x	
-Computerization				x	x	
-Coding					x	
-Machine					x	
-Numerical solver			(x)	x	x	
-Coding			(x)	x	x	
-Suitability			(x)	x	x	
-Mathematical method			(x)	x	x	
-Resolution			(x)	x	x	
-Parameters						

Figure 3 | Extension to the uncertainty matrix (parentheses stand for possible future levels of uncertainty).

The level of this uncertainty is now varying between total and recognised ignorance and with this work we want to raise its level to qualitative uncertainty.

Model coding uncertainty

Model coding uncertainty is caused by all programming errors that occur during the implementation of the conceptual model. It does not cover the coding uncertainty caused by the coding errors in the modelling framework that was used to develop the model. The identification of programming errors is usually referred to as code verification in general, and software quality assurance in particular. Most of the general techniques of software quality assurance (SQA), such as static analysis, coverage analysis, glass box, black box and regression testing, which are used in software development, can also be used to verify the code used to implement the conceptual model.

In the work of Zhigou *et al.* (1997) a quite different approach to identify and diagnose model coding errors has been presented. They use predicted feature matrices of the possible modelling errors, and design dedicated observers for the invalid models to generate a feature signal that is analysed to produce an indication of where the coding error is located in the model. The whole approach is based on two independent implementations of the same model and assumes that it is highly unlikely that the same coding errors are made in these two implementations.

In the work of Copp *et al.* (2008) it was found that getting five commercial simulation packages to produce exactly the same results for a benchmark model was extremely difficult, due to the presence of model errors that had to be corrected.

Of course, this source of uncertainty is epistemic. The NUSAP glossary also mentions this source of uncertainty under a more general term: "software error" (van der Sluijs *et al.* 2003).

Machine uncertainty

Machine uncertainty is a consequence of finite-precision floating point arithmetic and causes computer round-off errors, which always corrupt the results. The presence and accumulation of round-off errors depend on several factors.

Machine architectures can reserve different types of floating point precision (= the number of significant digits) for different types of results. For example, normal (e.g. four significant digits) precision can be reserved for initial values and final results, while long precision (e.g. six significant digits) is reserved for intermediate results. Furthermore, developers of model and programming language compilers have complete freedom in the process of associating data types (e.g. double, integer) to different types of precision. Even software libraries can provide their own manufactured precision types, which are the result of mathematical operations on the existing machine precisions.

Iterative processes, such as numerical integration can seriously cause accumulation of round-off errors. In the study of Goel & Dash (2007) on the weather forecasting model of the National Center for Medium Range Weather Forecasting it is shown that the difference between calculations on different machine architectures increases when the simulated time is increased to several months.

The NUSAP glossary partly handles this source of uncertainty by introducing a more general term: "hardware error", which is defined as all errors in model outcomes that arise from bugs in hardware. The accumulation of round-off errors is also mentioned in the NUSAP glossary as "numerical error" (van der Sluijs *et al.* 2003).

An interesting approach to quantify machine uncertainty is to apply some randomisation to floating point arithmetic and their operands, so that statistical analysis can be used to assess and predict round-off error accumulation. An overview of some techniques that use this approach is found in the work of Parker *et al.* (2000). They introduce the term "Monte Carlo arithmetic" (MCA) in which an inexact value is randomised as

$$\text{randomise}(x) = \begin{cases} x & \text{if } x \text{ is exact (within } t \text{ digits)} \\ x + 10^{\lfloor \log_{10}|x| \rfloor + 1 - t}\xi & \text{otherwise} \end{cases} \quad (2)$$

where t is the virtual precision and x is a random variable typically uniformly distributed over the interval $]-1/2,1/2[$. In this way every arithmetic operation on an inexact value is randomised in a predefined manner (see Table 2).

Table 2 | Example of randomization using eight-digit decimal arithmetic ($t = 8$) (Parker 1997)

Sampled value of ξ	$-0.14415412810232532...$
Value x	$+7.5111111$
Value of $10^{1-8}\,\xi$	$-0.00000001441541281023253 2...$
Randomise (x)	$+7.5111110855845871897674 68...$

Repeating an arithmetic operation on two inexact values results in a collection of slightly different values that comprise a sample distribution to which statistical analysis can be applied. These values differ only in the random digits of their errors ($10^{\lfloor \log_{10}|x| \rfloor + 1 - t}\,\xi$), for which the expected value is zero. This sampling distribution has a sample mean μ, which is an estimate of the exact result of the operation, a sample standard deviation σ, which estimates the error in one single result and a sample standard error $\Sigma/\Sigma n$, which estimates the error in the mean taken over n results. Calculations using this empirical approach can detect wrong results in cases where ordinary floating-point arithmetic is lacking. For example, if during an iteration the nth calculation has a σ that is as large as μ, instability can be a problem of the process under consideration. MCA can also detect catastrophic cancellation: this is the loss of leading significant digits caused by subtraction of two approximately equal values, where at least one of the values is inexact. Because MCA randomises the non-significant digits when the same subtraction is calculated repeatedly, these randomised digits will not reappear in the results while the remaining significant ones will (Parker *et al.* 2000).

The nature of this source of uncertainty is mainly due to a lack of knowledge and the limited capacity of the state-of-the-art technology, since from our perspective with the present and near-future technology round-off errors will not disappear.

Numerical solver uncertainty

This accumulated source of uncertainty considers all aspects of uncertainty that are related to the numerical solver, which is used to solve the equations of the conceptual model. This source of uncertainty encompasses words like "numerical error" and "inexactness" that are part of the NUSAP glossary (van der Sluijs *et al.* 2003). A numerical solver implemented in a programming language has an unknown or known range of problems to which it is applicable. Sometimes it may also calculate the error between the calculated value and an estimated asymptotic solution, obtained when the step size would be zero. At this moment the level of this uncertainty ranges from total to recognised ignorance, sometimes even to qualitative uncertainty, depending on the person who performs the simulations, in particular depending on his or her background and level of experience.

Unfortunately, the application of the most accepted methodologies for uncertainty assessment is not possible in this case, because these methods use statistical methods based on the use of probability distribution functions. These statistical methods are not applicable to all solver parameters, which often only have a few discrete values.

Two types of numerical solver uncertainty can be discerned: solver coding uncertainty and solver suitability uncertainty.

Numerical solver coding uncertainty

Numerical solver coding uncertainty is restricted to all programming errors that can occur in the implementation of the numerical solver. When the solver implementation is separated from the model coding implementation, this uncertainty is present as a separate source and all techniques of SQA can be applied to it. The responsibility for this type of SQA lies with the software developers of the numerical solver code.

If the code of the numerical solver is intertwined with the code of the implementation of the conceptual model, numerical solver coding uncertainty and model coding uncertainty could be lumped together into a broader term: coding uncertainty. SQA should then be applied by the model developer.

When the implementation of the numerical solver is lumped with the code of the entire modelling framework, this separate source of uncertainty still exists, but the responsibility of SQA then lies with the developers of the modelling framework.

Numerical solver suitability uncertainty

It is a well-known fact that a numerical solver, although suitable for a certain type of differential Equations (PDE,

ODE, DAE, etc.), cannot correctly compute the solution of every possible conceptual model constructed with that type of differential equation. When the chosen solver and its settings are appropriate, it will not influence the behaviour of the solution significantly, but when it is incorrect the influence becomes bigger, and so does the total uncertainty of the model simulation. As an example, we would like to point out that the correctness of numerous methodologies used for assessment of other sources of uncertainty is influenced by numerical solver uncertainty. Moreover, numerical solver uncertainty in general, and numerical solver suitability uncertainty in particular, have a major impact on the computation time: choosing a suitable numerical solver can significantly speed up calculation time (Claeys *et al.* 2007).

Numerical solver suitability uncertainty can be divided into a part that relates to the uncertainty caused by the interaction between the mathematical method (mathematical method uncertainty) and the model's properties (e.g. stiffness) and a part that applies to the discretisation error, which depends on the step size (resolution uncertainty). An explanation of these two categories follows.

Stiffness is one of the properties of the solution trajectory that can cause mathematical method uncertainty. Scientists often describe a system as stiff if the rate of change of the derivable state variables differs widely amongst the derivable state variables. Stiffness depends on the differential equations themselves, on the accuracy chosen by the user, on the length of the integration interval and on the region of absolute stability of the method. Several stiffness detection mechanisms exist (Ascher & Petzold 1998; Cameron *et al.* 2001), but for these the eigenvalues of the Jacobian matrix are needed, and for nonlinear systems these eigenvalues strongly depend on the state of the system which in turn varies with the independent variable (mostly time). An illustration of these varying eigenvalues is given in the work of Steffens *et al.* (1997) and in that of Seppelt & Richter (2005) for the predator–prey model (see Equation (1)). Moreover, for many stiffness tests symbolic manipulations are needed, for which extra computation time is required, especially in the case of environmental models, that can contain huge systems of ODEs. Solvers that are suitable to solve stiff problems, according to their documentation, are: LSODE/BDF/Newton/Dense (Hindmarsh 1983), CVODE/BDF/Newton/Dense (Hindmarsh *et al.* 2005), DASRT (Brenan *et al.* 1989), DASSL (Petzold 1983*a*) and LSODA (Petzold 1983*b*).

For resolution uncertainty the discretisation error can be assessed using numerical error estimation. However, a solution trajectory can diverge from the real solution trajectory if the choice of the step size is too large for the numerical method used. In this case instability occurs and some numerical solvers will not converge. Hence the computation can fail, giving rise to a convergence error or can generate wrong results, without failure. Typically a limit on the step size, which ensures convergence, exists for every combination of the conceptual model, its initial values and a numerical method (Ascher & Petzold 1998). This source of uncertainty is mainly epistemic, because it can be minimised or bounded by further research, and development of new technologies. Some numerical solvers can detect this type of uncertainty by testing convergence and returning error messages, when convergence fails (e.g. CVODE (Hindmarsh *et al.* 2005) and LSODE (Hindmarsh 1983)).

The suitability of the numerical solver can contribute significantly to the total uncertainty of the model simulation. In order to reduce numerical solver suitability uncertainty, future versions of modelling frameworks must focus on guidance to aid the user with the choice of an appropriate numerical solver and its settings. Research regarding automatic selection of numerical solvers and their settings has been reported (Claeys 2008).

CONCLUSIONS

Basically, any method that involves the numerical computation of the solution of differential equations is influenced by numerical solver uncertainty. For this reason, this source of uncertainty can be as important as parameter and model structure uncertainty, as shown in the illustrative example.

It is clear that numerical solver uncertainty should not be ignored. Future research is needed to assess and reduce this source of uncertainty. The introduction of a new terminology to support scientific communication on this subject is a first step in a long journey towards better assessment of this type of uncertainty. The goal is to raise the level from total or recognised ignorance to what is termed qualitative uncertainty.

REFERENCES

Abramowitz, M. & Stegun, I. A. 1965 *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. Dover, New York.

Ascher, U. M. & Petzold, L. R. 1998 *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*. SIAM, Philadelphia, PA.

Benedetti, L., Bixio, D., Claeys, F. & Vanrolleghem, P. A. 2008 Tools to support a model-based methodology for emission/immission and benefit/cost/risk analysis of wastewater systems that considers uncertainty. *Environ. Modell. Softw.* **23**, 1082–1091.

Brenan, K. E., Campbell, S. L. & Petzold, L. R. 1989 *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations*. SIAM, Philadelphia, PA.

Butcher, J. C. 2003 *Numerical Methods for Ordinary Differential Equations*. John Wiley & Sons, Chichester.

Cameron, I., Hangos, K., Stephanopoulos, G. & Perkins, J. 2001 *Process Modelling and Model Analysis*. Academic, London.

Claeys, P. 2008 *Towards Automatic Numerical Solver Selection for Hierarchical Bioprocess Models*. PhD Thesis, Ghent University, Belgium.

Claeys, F., Vanrolleghem, P. A. & Fritzson, P. 2006 A generalized framework for abstraction and dynamic loading of numerical solvers. In: *Proc. 2006 European Modelling and Simulation Symposium (EMSS), Barcelona, Spain, 4–6 October*.

Claeys, P., De Baets, B. & Vanrolleghem, P. A. 2007 Towards automatic numerical solver selection using a repository of pre-run simulations. In: *Proc. 7th International IWA Symposium on Systems Analysis and Integrated Assessment in Water Management, Washington, DC, 7–9 May* (on CD-ROM).

Copp, J. B., Jeppsson, U. & Vanrolleghem, P. A. 2008 The benchmark simulation models: a valuable collection of modelling tools. In: *Proc. International Congress on Environmental Modelling and Software (iems2008), Barcelona, Spain, 7–10 July*.

Flowers, B. H. 2000 *An Introduction to Numerical Methods in C++*, Revised edition. Oxford University Press, Oxford.

Freitas, C. J. 2002 The issue of numerical uncertainty. *Appl. Math. Modell.* **26**, 237–248.

Goel, S. & Dash, S. K. 2007 Response of model simulated weather parameters to round-off-errors on different systems. *Environ. Modell. Softw.* **22** (8), 1164–1174.

Hairer, E., Nørsett, S. P. & Wanner, G. 1993 *Solving Ordinary Differential Equations I: Nonstiff Problems*. Springer-Verlag, Berlin.

Hindmarsh, A. C. 1983 ODEPACK: a systematized collection of ODE solvers. In *Scientific Computing*, (ed. in R. S. Stepleman, *et al.*), pp. 55–64. North-Holland, Amsterdam.

Hindmarsh, A. C., Brown, P. N., Grant, K. E., Lee, S. L., Serban, R., Shumaker, D. E. & Woodward, C. S. 2005 SUNDIALS: suite of nonlinear and differential/algebraic equation solvers. *ACM Trans. Math. Softw.* **31** (3), 363–396.

Kloeden, P. E. & Platen, E. 1995 *Numerical Solution of Stochastic Differential Equations*. Springer-Verlag, Berlin.

Oberkampf, W. L., Trucano, T. G. & Hirsch, C. 2004 Verification, validation, and predictive capability in computational engineering and physics. *Appl. Mech. Rev.* **57**, 345–384.

Parker, D. S. 1997 *Monte Carlo Arithmetic: Exploiting Randomness in Floating-point Arithmetic*. Available at: http://www.cs.ucla.edu/stott/mca/CSD\-970002.ps.gz

Parker, D. S., Pierce, B. & Eggert, P. R. 2000 Monte Carlo arithmetic: how to gamble with floating point and win. *Comput. Sci. Eng.* **2** (4), 58–68.

Petzold, L. R. 1983*a* A description of DASSL: a differential/algebraic system solver. In *Scientific Computing*, (ed. in R. S. Stepleman, et al.), pp. 65–68. North-Holland, Amsterdam.

Petzold, L. R. 1983*b* Automatic selection of methods for solving stiff and nonstiff systems of ordinary differential equations. *SIAM J. Sci. Stat. Comput.* **4** (1), 136–148.

Press, W. H., Teukolsky, S. A., Vetterling, T. & Flannery, B. 1992 *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, Cambridge.

Refsgaard, J. C., Van der Sluijs, P., Højberg, A. L. & Vanrolleghem, P. A. 2007 Uncertainty in the environmental modelling process–a framework and guidance. *Environ. Modell. Softw.* **22** (11), 1543–1556.

Salari, P. & Knupp, P. 2000 *Code Verification by the Method of Manufactured Solutions*. SAND 2000-1444. Sandia National Laboratories, Albuquerque, NM.

Seppelt, R. & Richter, O. 2005 It was an artefact not the results–a note on system dynamic model development. *Environ. Modell. Softw.* **20**, 1543–1548.

Shampine, L. F. 1982 Implementation of Rosenbrock methods. *ACM Trans. Math. Softw.* **8** (2), 93–113.

Steffens, M. A., Lant, P. A. & Newell, R. B. 1997 A systematic approach for reducing complex biological wastewater treatment models. *Water Res.* **1** (3), 590–606.

van der Sluijs, J. P., Risbey, J. S., Kloprogge, P., Ravetz, J., Funtowicz, S. O., Quintana, S. C., Pereira, A. G., De Marchi, B., Petersen, A. C., Janssen, P. H. M., Hoppe, R. & Huijs, S. W. F. 2003 *Guidance for Uncertainty Assessment and Communication: Detailed Guidance* Vol. 3. *Strategic Research Project Uncertainty Analysis* S/550002. RIVM/MNP, Utrecht University.

van Griensven, A. & Meixner, T. 2004 Dealing with unidentifiable sources of uncertainty within environmental models. In: *Proc. International Environmental Modelling and Software Society, University of Osnabrück, Germany, 14–17 June*.

Walker, W. E., Harremoes, P., Rotmans, J., Van der Sluijs, J. P., van Asselt, M. B. A., Janssen, P. & Krayer von Krauss, M. P. 2003 Defining uncertainty: a conceptual basis for uncertainty management in model-based decision support. *Integr. Assess.* **4** (13), 5–17.

Zhigou, Y., Vanrolleghem, P. A. & Vansteenkiste, G. C. 1997 Modeling error identification of activated sludge models. *Water Sci. Technol.* **36** (52), 81–88.

First received 13 October 2008; accepted in revised form 5 June 2009. Available online 5 December 2009