Exploration of Uncertainty in Glacier Modelling

David E. Thompson
NASA Ames Research Center, M.S. 269-3, Moffett Field, CA 94035

ABSTRACT. There are procedures and methods for verification of coding algebra and for validations of models and calculations that are in use in the aerospace computational fluid dynamics (CFD) community. These methods would be efficacious if used by the glacier dynamics modelling community. This paper is a presentation of some of those methods, and how they might be applied to uncertainty management supporting code verification and model validation for glacier dynamics. The similarities and differences between their use in CFD analysis and the proposed application of these methods to glacier modelling are discussed. After establishing sources of uncertainty and methods for code verification, the paper looks at a representative sampling of verification and validation efforts that are underway in the glacier modelling community, and establishes a context for these within overall solution quality assessment. Finally, an information architecture and interactive interface is introduced and advocated. This Integrated Cryospheric Exploration (ICE) Environment is proposed for exploring and managing sources of uncertainty in glacier modelling codes and methods, and for supporting scientific numerical exploration and verification. The details and functionality of this Environment are described based on modifications of a system already developed for CFD modelling and analysis.

INTRODUCTION AND BACKGROUND

Verification of numerical glacier models generally intends to answer the question of whether the governing equations were solved correctly. That is, does one believe that the process of solving a model of discretized equations plus their boundary conditions, initial conditions, any input data, and conceptual modelling assumptions has yielded a converged solution that is correct within bounded numerical error. Verification arises by assessing numerical accuracy against expectations of the theoretical model, and by strict uncertainty modelling and analysis of the model parameters, limits, and numerical convergence histories of the code over specified grids. Usually, the problem statement itself should indicate what counts as an acceptable level of numerical accuracy and solution stability by defining the purpose and fidelity of the analysis. Accuracy is measured against the scientists' codified expectations from this problem statement. Expectations arise from a variety of sources: field observations and data, satellite sensor data, and laboratory experiments, each of which are incomplete data-sets that approximate reality; benchmark analytic solutions intended to exercise the code; code performance across a variety of grids; parameter resolution studies and sensitivity-uncertainty analyses; and the matching of results to previously verified solutions and methods on "similar" computational problems within a specified tolerance. The model's ability to match expectations can constitute a verification of that model as long as the sources of numerical error are understood and can be accounted for in terms of technical analyses. Such analyses includes identification of both computational and flow physics uncertainties, formal discretization order and accuracy, solution stability during grid convergence studies, and general robustness of the numerical method.

However, there exists a distinction between verification of a code or model (the numerical accuracy) and validation of the model (the correct conceptual problem statement), and uncertainty and errors can arise from both sources. Here again, the problem statement itself will dictate what counts as a solution by projecting expected results or solution behavior. Consider the process of recovery from mismatched results. If data and model results do not agree, either one decides that the model is accurate and then analyzes why the data may be faulty, incomplete, inconsistent, or not fully applicable (data errors); or one decides that the data are accurate and hence the model results must be numerically wrong. If the model is wrong, the problem may either be improper discretization or non-convergence over the chosen grid (verification errors) and the numerical scheme must be reformulated; or the model may be an inaccurate representation of the physical situation of interest (validation errors). In that case, the equation results will never match the observed data, and the model itself must be reformulated to capture the correct physics. We may be solving the equations correctly, but we are solving the wrong set of equations. Note that an <u>incomplete</u> representation of the physics is not by itself a validation error or a wrong set of equations. It may well be an intentional model simplification or approximation in order to isolate and study certain processes or to maintain numerical tractability. But, an incomplete representation <u>will</u> induce uncertainties.

Verification and validation of codes and models are thus distinct processes. However, since the overall goal is to establish the credibility and solution quality of the numerical model, to match the conceptual model to reality, and to identify and manage the uncertainties, these two processes are frequently employed together to these ends. In fact, the two processes may be intertwined whereby validation establishes the context and content of the physical problem being modelled and dictates levels of acceptability and uncertainty, and verification establishes the numerical accuracy of the codes and mathematical constructs through rigorous methods. The two processes together create credibility, confidence in solution quality, and insight and intuition for further exploration.

In 1994, Oreskes and others published an article in Science entitled "Verification, Validation, and Confirmation of Numerical Models in the Earth Sciences." In this paper they argue, from a philosophical point of view, that verification and validation of numerical models in the earth sciences is impossible due to the fact that natural systems are not closed, so we can only know about them incompletely and by approximation. Although surfacing significant philosophical issues for verification, validation and truth in numerical models, unfortunately their arguments to support this thesis fail frequently due to actual conflation and misuse of the very terms and concepts the paper is intended to elucidate. They correctly argue that numerical accuracy of the mathematical components of a model can be verified. They also hold that the models that use such mathematical constructs (the algorithms and code) represent descriptions of systems that are not closed, and therefore use input parameters and fields that are not and cannot be completely known. But this is not a successful argument against the possibility of model verification; rather, it is an issue of whether the application of a numerical model appropriately represents the problem being studied. Hence, this is a problem in model validation or model legitimacy, not one in verification.

In terms of model validation, Oreskes and others (1994) discuss establishing the "legitimacy" of the model as akin to validation. As far as it goes, this representation is accurate; but they claim further that a model that is internally consistent and evinces no detectable flaw can be considered as valid. This is a necessary, but not sufficient, condition for validation of models, and it derives from the philosophical usage of establishing valid logical arguments, not from practices in computational physics. Hence, once again their arguments are using terms from the philosophical literature that carry different technical meaning and reference from those same terms in the scientific literature. Such misuse is clear when they say that it is "misleading to suggest that a model is an accurate representation of physical reality" (p.642). In point of fact, the intent of the scientific model is to represent reality or a restricted and defined simplification of physical processes, and validation is specifically the process that demonstrates the degree to which the conceptual model (the mathematical representation) actually maps to reality. The fact that a model is an approximation to reality does not mean such representation is "not even a theoretical possibility" (p.642) due to incompleteness of our data-sets. Rather, such analysis and mapping dictates exactly what validation of the model means, and what the limits of applicability of the model are. Establishing validation means establishing the degree to which the conceptual model is even supposed to encompass physical reality.

Verification, validation, and recognition and control of uncertainties are precisely the way the scientific community deals directly with the acknowledged problem of open systems and incomplete data at a variety of scales. There are certainly philosophical issues to be addressed in verification, validation, and uncertainty management for dynamical systems modelling, such as choice of closure of equations in analysis and representation in simplified or reduced models. But numerical accuracy and verification of models is not about philosophical truth or common usage of terms. Credibility of simulations is established by recognizing computational and flow physics uncertainties and by quantifying them. The scientific community properly deals with incompleteness and openness of natural systems through quantification of model limits and ranges of applicability, through model representation alternatives, through specific numerical accuracy tests, through analyzing the scale of coupling of various linked flow processes, through sensitivity and uncertainty analysis, and so on. Once "natural philosophy" became rigorous and computational, then verification, validation, and uncertainty management in science codes and science strategy scenarios is certainly not only possible but necessary. And whereas understanding numerical accuracy of codes is important for credibility, understanding model limitations is essential for model applicability and use in science problems.

The aerospace computational fluid dynamics (CFD) community has been engaged in substantial efforts at defining and quantifying uncertainty, and developing verification, validation, and code calibration processes since the mid-1980's (Mehta, 1991, 1998; Roache, 1997, 1998). This paper presents some of these methods, and clarifies the similarities and differences between their use in CFD analysis and proposed application of these methods to glacier modelling. First, the sources of uncertainty and errors are discussed. Then appropriate procedures for validation and verification are presented. Next, the paper looks at a representative sampling of verification and validation efforts that are underway in the glacier modelling community, and establishes a context for these papers within overall solution quality assessment. Finally, an information architecture is introduced and advocated. This Integrated Cryospheric Exploration (ICE) Environment is proposed for supporting scientific numerical exploration and for exploring and managing sources of uncertainty, whether from data-set, verification, or validation errors. The details and functionality of this Environment are described based on modifications of a system already developed for CFD modelling and analysis.

SOURCES OF UNCERTAINTY IN MODELS AND CODES

Uncertainty refers to a perceived lack of reliability or confidence in the results of a simulation of some physical process. Uncertainty specifically refers to errors, but not to mistakes. Errors are expected to arise simply from the fact that the physical process of interest is being modelled by continuum partial differential equations (PDEs) and their boundary conditions, and from the fact that the continuum model is then re-represented in discrete mathematics for machine implementation. Identifying and quantifying uncertainties, and understanding their origins and propagation through both the conceptual and numerical models and codes, is done to establish credibility in these representations of reality.

It is very important to separate sources of errors and uncertainties that arise from the representation of the physical process as a mathematical model from those that arise from the discretization of the math model and its operation by the computer. For example, two sources of error in a code may interact or cancel each other at some scales and resolutions thereby hiding real features, or non-physical behavior may arise in the numerical process that is then masked. Alternatively, changes that are made for convenience to a boundary condition in the code must be evaluated as to whether this is a reasonable boundary condition in the continuum model, or non-physical simulations may result even though the numerical model converges. Such interactions and changes must be tracked and isolated so that actual uncertainties can be measured. This desire has led to separating the model validation process from the code verification process. Thus in glacier modelling, one must be separately concerned with errors and uncertainties arising from the flow physics representations, with those strictly due to the numerical scheme, and with the propagation of errors that are associated with local features in model or code into more regional or global parts of the model. Local errors and uncertainties are frequently accessible to analysis by algorithmic methods. But as such uncertainties propagate through the model evolution, or impact upstream or downstream of their source, the tracking of such uncertainties becomes a problem in model interpretation that is frequently not amenable to algorithmic analysis. Strategies to explore impacts and propagation of errors must be worked out based on target results and variations of these.

In this section, the sources of errors and uncertainties that arise in glacier models and their computational flow codes are discussed. The basis for this discussion and categorization of features and errors is drawn from the development of uncertainty identification and management that has been occurring recently in the CFD community (Mehta, 1991, 1998; Oberkampf and Blottner, 1998; Roache, 1997, 1998). Clearly, both in use and focus, CFD for aerospace development and design is rather different from that of fluid dynamics for glacier and ice sheet modelling. However, the lessons learned in the CFD community can be mapped into expectations and guidance for development of verification and validation scenarios by the glacier modelling community. Furthermore, the structure of uncertainty analysis between aerodynamics and glacier modelling can be remarkably similar. Both disciplines start their analysis from the Navier-Stokes equations for momentum balance; they both include auxiliary process models that become proxy for and replace simple boundary conditions; and the failing of theorems about equivalence between continuum and discrete models arises due to the nonlinearity of the PDEs rather than just presence or absence of various terms like inertial or viscous relations that normally distinguish glacier flow from aerodynamic analysis.

Sources of Modelling Uncertainty

Uncertainties in creating a mathematical model of a physical process are caused by inaccuracies, approximations, and assumptions in the mathematical representation. These errors are completely distinct from numerical errors that will be discussed later. Such inaccuracies may be intentional in that we know an approximation to the physics has been made to ensure mathematical tractability or from the desire to isolate certain processes. However, the extent of the inaccuracies, or their impact on model evolution and representation, may not be fully anticipated. Other inaccuracies arise unintentionally due to our paucity of understanding of the details and interaction scales of the processes being modelled. In any case, these inaccuracies can be categorized as to source if not effect or extent. Part of exploration of these uncertainties and their propagation is to develop better scientific understanding, intuition for future refinements, and for general model validation through comparison to data from experiments (physical or computational), from field work on glaciers or deglaciated beds, and from satellite sensors.

Consider first the uncertainties that arise from the PDEs of fluid dynamics. There is often not a complete understanding of the phenomena or processes being modelled -- in fact, this is usually the reason for doing the simulation. Hence, the mathematical representations will be incomplete. The modelling parameters may be incompletely known or be intentionally averaged. There may be a lack of field data for describing expected ranges of these parameters or other dependent variables. In short, the math model is generally a simplified picture of reality. The uncertainties related to this simplification depend heavily on whether the mathematical representation has captured the dominant physics at the right scale for the process being investigated. To guarantee control or understanding of the uncertainties, frequently the PDEs are formulated so that the simplest appropriate forms of the equations can be used. Then the equations are extended to more complicated flow situations in a reasoned manner. But as the model is extended, more accurate information about the processes is required, and hence more uncertainty may be introduced. Errors occur at each level of approximation.

The model may be written to isolate features or processes of interest, and these are then uncoupled from real system behavior. The purpose of isolation may be to better understand the process or simply that it is not feasible to model everything simultaneously. However, such isolation implicitly assumes that either there is no influence between the process and the rest of the system, or that the influence is known and understood. The degree to which this is untrue introduces uncertainties into the PDE model and its final results.

Oberkampf and Blottner (1998) describe a chain of increasing complexity of flow problems in which more detailed physics is added during CFD analysis. Usually the aerodynamicist starts with inviscid equations. One then progresses to full viscous

Navier-Stokes equations using the inviscid results as guidance, although this can give misleading results. A variety of increasing approximations and complexity directions can follow. One can analyze mixed species flows. One can include turbulence models for recognizing and simulating particular features in certain flow configurations, namely types of shock structures or vortex cores. Alternatively, one might add or couple together specific physical phenomena with the general flow field, or perhaps use transition models that cross both laminar and unsteady flow conditions through changing temporal and spatial characteristics.

In glacier modelling, a parallel series of increasing complexity might be to start as normal with restricted viscous flow but with complicated rheology. Then one might add thermodynamic models. Our mixed species equivalent could be modelling entrained debris at the ice-bedrock interface or till deposition models coupled to the ice flow. If a true coupling of ice flow and basal processes are modelled, this requires additional algebraic relations or perhaps new PDEs rather than just specified boundary conditions (Blatter and others, 1998). Next one might envision developing models for transient phenomena. For example, one model might assess the transient flow characteristics that arise as a glacier evolves from one steady state to another. A second model might study the growth to finite amplitude of secondary flow structures in the ice. These would require solution of additional PDEs. Then, rather than using averaging methods or scaling of governing equations, one might couple these specific transient phenomena into the general flow field. Increasing complexity leads to increasing understanding, but also to increasing uncertainty in the PDEs that must be evaluated.

Modelling may inadvertently introduce extraneous physical (or computational) phenomena and features due to the perturbation and constraint of existing flow conditions from use of simplification. For example, a 2-dimensional representation automatically eliminates the possibility of modelling 3-dimensional features like vortices or even simple transverse flow. That simplification is understood and frequently made intentionally. But restricting flow to 2-dimensions also introduces extraneous constraints at boundaries unless complicated flux boundary conditions are created to prevent over-constraining the flow that could introduce non-physical instabilities or discontinuities. The ramifications of such assumptions must be thoroughly explored and understood in order to maintain credibility of the resulting flow solutions.

Uncertainties can arise separately in the auxiliary physical models as well, some of which themselves may be PDEs. Generally, auxiliary physical models refers to equations needed to close the momentum balance equations so that a solution is possible. In simplified CFD, this usually means employing equations of state approximations and thermodynamic parameters or relations. As the CFD equations increase in complexity, the auxiliary relations include transport properties and constitutive relations between stress and strain-rate in order to model eddy viscosity. The flow may also be coupled to structural bending due to pressure loads over a wing. Also, new turbulence models are used whose ranges of applicability must be determined by experimental results (usually in wind tunnels) to resolve rapidly varying flow conditions.

In the glacier modelling analog, our auxiliary relations are the constitutive relations and flow laws from which effective viscosity profiles are derived. We may include thermodynamic relations and heat flux models that require additional energy balance equations coupled to the flow equations. An asymmetry occurs in glacier modelling in that the equations are initially written requiring stress tensor derivatives. Laboratory tests on ice samples yield strain-rate data, and field measurements on glaciers yield velocity and strain-rate data. Viscosity is derived by modelling this data with flow law assumptions, and the governing equations are then written either in terms of velocities, viscosities, and their derivatives, or in terms of velocities and strain-rates with viscosity becoming an explicit function of strain-rates through the assumed flow law. Alternatively, the equations may also be written entirely in terms of ice thickness and fluxes with the flow law as an auxiliary constraint. Thus, these auxiliary relations and transformations lead to highly nonlinear PDEs even for simple flow fields.

Finally, sources of modelling uncertainty arise from the representation of boundary conditions (BCs). Dirichlet conditions that specify velocity and temperature offer no problems; but any BC that specifies more than that requires information from the interior domain of the flow. Generally the flow solution must be found, then that solution is used in a separate PDE as a consistency condition, and this new equation is solved. Selecting the correct matching condition or consistency relation is necessary to control the uncertainty in the model. Additionally, BCs along a glacier bed may require boundary discontinuities in velocity or stress, accurate or scaled representations of bedrock geometry, and analysis of the fidelity of the computation over that geometry (see for example, Colinge and Blatter, (1998) and Blatter and others (1998), for problems identified and the impact at the numerical level). One obvious discontinuity comes from representing the flow field at glacier margins, and uncertainties arise related to selecting an adequate resolution of conditions at these moving or fixed margins. Mathematical singularities may be handled in the continuum mathematics; but with the ultimate goal of deriving numerical simulations, such singularities may be very difficult to program numerically. Hence, the PDEs and their discrete representations will turn out to be different problems.

Uncertainty in free surface conditions generally arises from the problem of selecting the correct matching conditions to apply based on the phenomena to be analyzed. For example, a simple claim of continuity or vanishing normal velocity or shear stress at a deformed surface may not be sufficient if actual mass transport is anticipated across that surface to induce flow perturbations from an accumulation event. Interface conditions are equally important; for example, one may decide that heat flow is continuous across the interface but perhaps a simple match of temperatures will be sufficient for the case at hand. Numerical modelling schemes may also introduce layers of interfaces across which constraints against discontinuities are imposed.

The most difficult conditions are probably the open BCs. These include inflow/outflow conditions in CFD, and include flux conditions in glacier modelling. Examples would be specifying coupling or transition conditions from an ice divide into channelized flow, ice sheet coupling to ice streams (Marshall and Clarke, 1996), or modelling till deformation coupled with entrainment or deposition from the ice flow field. The issue is how to specify them while maintaining control on uncertainty, and often where to apply them (at infinity margins or near features of interest, like ice rises). If the PDE is over-constrained by these conditions, the solution may exhibit instabilities that contaminate the actual flow solution; if the flux conditions are not coupled properly, the resulting solution is spurious.

The overall lesson is that developing the mathematical model requires identification and management of uncertainties. These uncertainties are what must be expunged, mitigated, or accommodated during the model validation process, whether results are compared with lab or field data or with previously benchmarked flow solutions that act as test cases for validation.

Sources of Code and Numerical Uncertainty

Code and numerical uncertainty leads directly to issues of verification as opposed to validation. Generally, all sources of errors in codes and calculations arise either from questions of equivalence of the discrete computational model to the original mathematical PDE model, or from questions of the numerical accuracy of the discretization scheme and its implementation. In this section, these two sources of uncertainty and error are examined.

Roache (1998) outlines five sources of error in code development and use. Code authors can make errors during code generation and while developing code use instructions. Code users can make errors in their problem set-up, in their definition and coding of benchmark cases (possibly analytic) for use in results comparison, and in their interpretation of code results. Hopefully, the code verification process would remove the error source derived from code and instruction development. Code use errors however may arise anytime a (verified) code is applied to a new problem statement. When a code is applied to a new problem, the fact that the code has been previously verified does not give any estimate of its accuracy for this new problem. Systematic grid convergence studies are needed to verify the code on the new problem domain, and this is separate from the activity of validation that the PDEs are appropriate for the new problem.

Consider the problem of code and instruction generation. The solution procedure used in an algorithm or code is an approximation to the original PDEs. Errors arise in coding due to the discretization scheme selected that is intended to map the PDE model into a finite difference, finite element, or finite volume representation. Thus, the PDEs, the auxiliary equations, and the boundary conditions are all discretized to some order, defined by a truncation error of the series solution. The truncation error allows one to define the order of accuracy of the solution; the discretization error gives the numerical error of the calculation due to the fact that the solution is sought over a finite number of grid points. In addition to discretization errors, there can also be errors in the computed solution of the set of discretized equations representing the PDEs, and these are not necessarily related to grid convergence (the discretization accuracy), or to the numerical mapping and establishment of order of accuracy. These additional errors arise from the behavior of the numerical scheme itself.

The foundation for the discrete math approach to solving PDEs numerically is based on an equivalence theorem (Oberkampf and Blottner, 1998) that says: 1) the difference equations are equivalent to the differential equations (guaranteeing consistency); and 2) the solution remains bounded as it is solved iteratively over the space and time discretizations Δ (guaranteeing stability). The problem for the PDEs of fluid dynamics and of glacier modelling is that this theorem only provides necessary and sufficient conditions for convergence for linear PDEs. For nonlinear problems, consistency and stability are only necessary conditions (not sufficient) that the numerical solution will converge to the continuum solution in the limit of infinitesimal Δ. Hence, the numerical solution may oscillate, diverge, or only converge slowly and perhaps to an alternate solution or spurious steady state. Hindmasrh and Payne (1996) introduce the use of iterated maps as a way of following the evolution of the numerical solution so as to understand its reasons for oscillations or spurious convergences. So, in terms of practical equivalence, how the difference equations are written and solved can determine what solution flow field is obtained. This behavior is especially apparent in either over- or under-specification of the discretized boundary conditions, but it also has to do with the residual accuracy of the calculation -- sometimes thought of as the speed of iteration to an acceptable convergence. However, Roache (1997, 1998) makes the point that grid generation errors or the construction of bad grids will add to the size of the discretization error, but they do not add new terms to the error analysis. So as the discretization improves, all errors that are ordered in Δ will go to zero, and hence an error taxonomy should not include both grid and discretization errors as separate categories. This is because one cannot take the limit of infinite order (equivalent to zero truncation error) without also taking the limit of infinite number of grid points. Yet in practice, the code developers may include iterative tuning or relaxation parameters, that are problem dependent, to speed iterative convergence, and this may contaminate the actual grid convergence (discretization accuracy) results.

Equivalent sources of error can arise from the discretization of the auxiliary relations and the boundary conditions. If the auxiliary equations are linear algebraic expressions, the formal error is equivalent to the machine round-off error. But since many of the auxiliary relations for our PDEs are nonlinear expressions, then some iterative technique is required for their solution and coupling to the momentum equations. Any errors that arise in any iterative step that is not fully converged will propagate through the solution process. In glacier modelling, such errors might arise when modelling equilibrium chemistry at the glacier bed, or transport and deformation of substrate materials. Many errors can be reduced by good interpolation schemes, but one needs to know the required accuracies of the problem in order to impose interpolations that do not induce uncertainties. In particular, the errors in the approximate representation of the properties and features being studied need to be on the same order as the errors introduced by the approximation techniques for interpolation, or else the numerical model cannot resolve the features. In CFD, the turbulence models used must be appropriate for the conditions and flight configurations being simulated. In glacier modelling, an auxiliary model of preferred fabric orientation that might enhance flow to surge status may not be appropriate as an auxiliary model, even if physically accurate, due to the disparity between scale of resolution of the flow model to ice fabric scales. Rather, a model would need to be created that mapped changes in fabric to changes in strain-softening in the effective viscosity. This 3-dimensional viscosity model would act as a proxy for material nonlinearities, coupling fabric orientation and stress gradients, and it would operate in the momentum balance equations as an auxiliary relation.

The discretized boundary conditions must provide consistent information for the solution of the discretized PDEs. According to Oberkampf and Blottner (1998), the balance between over- and under-specification of knowledge on the boundaries of the finite-difference model is more difficult to obtain and implement than is this information for the original PDEs. Recall that over-specification of discrete BCs can cause divergence in the numerical scheme, and under-specification can cause lack of convergence, solution wandering, or convergence to alternate steady states depending on grid sizes, features to be resolved, and relaxation parameters used. They speculate that this additional difficulty of matching BCs in the discrete case is due to the fact that the particular differencing scheme and the grid size determines the degree of coupling of the BCs to the interior flow equations. Conversely, in continuum math, the PDEs are always fully coupled to the boundaries. Thus sources of error and uncertainties need to be monitored and explored to maintain credibility of the model verification results. Only a rigorous grid refinement study will establish the overall order and accuracy of the complete numerical scheme (equations, auxiliary relations, and BCs) and will provide confidence in matching accuracy and spatial differencing scales. Grid convergence methods are discussed in the following section.

Additional representation complications, and hence sources of numerical error and uncertainty, arise from the cases of flux BCs and the use of unsteady or moving BCs. As mentioned previously, flux BCs are difficult to select and implement in the continuum PDE case, and in the discrete case these problems remain even for steady laminar flow. Oberkampf and Blottner (1998) site CFD cases wherein there is clash between numerical solutions and experimental data. When comparing the analytic limit of $\Delta \longrightarrow 0$ for the inflow/outflow BCs, these were found to be incompatible with the original PDEs (p.693). For cases of unsteady or moving BCs, the CFD example is one of analysis of reacting flows. For the glacier analog, instead of a specific BC for basal conditions, we might envision process models as consistency conditions that are used to describe entrainment of till and general bed deformation. Such complications are realistic, but are difficult to model and match to the glacier flow field. One way to gain confidence in these processes is to numerically experiment with the sensitivity of the flow to variations in these process models. One could then eventually extract a representative moving boundary relation; or one might use the ice "ceiling" as a BC for a basal deformation flow model and thus circumvent moving boundaries.

Finally, consider Roache's (1998) error categories that pertain to code use, namely problem set-up, coding of benchmark cases, and interpretation of results. These all refer to errors and uncertainties in the final discrete solution and results comparisons. It is important to note that to minimize solution errors, the magnitude of change tolerated on the dependent variables during iterations depends on the time step limit (Hindmarsh and Payne, 1996; Oberkampf and Blottner, 1998) as well as on the current convergence rate. One wants the iterative convergence error (that is, the residual accuracy) to be smaller than the temporal discretization error before going to the next time step. Hence one needs to compute the residual for each equation at each grid point, and let this residual approach some bounded value for all difference equations at all grid points. This will guarantee control of the discrete solution errors, and is necessary for stability. It is not always sufficient to merely iterate a solution until there are small changes in the dependent variables; non-movement of dependent variables does not guarantee small solution error. Solution strategies and their specific implementations must be explored to gain appreciation for the variability of discretization schemes and system performance. This knowledge lends credibility to target solution results.

PROCEDURES FOR VALIDATION AND VERIFICATION

The strategy for carrying out validation and verification processes is iterative. The outcome of model validation is meaningless without identifying and understanding the effects of numerical errors and their propagation on the model. This would say that verification should proceed validation. But part of validation is establishing the correct problem statement, the best representation for that problem in various sets of continuum PDEs and their boundary conditions, and an assignment of parameter values or ranges. This needs to be done before the discretized PDEs are formulated, the numerical scheme and gridding is selected, the code is calibrated across test cases in numerical experiments, and the scheme is verified. Thus, the process is iterative, and refinement of fidelity of both the physics and its numerical representation must proceed step-wise.

This section first examines validation methods that establish the context and content of problem statements. It is followed by presentation of some rigorous numerical methods, that have been developed and used for incompressible Navier-Stokes and related PDEs, and that can be used to verify the numerical accuracy and limits of such discretized codes. In this sense, verification is the process of doing experiments on the numerical schemes themselves, and this is different from the process of benchmarking the codes to test cases that represent a certain fidelity of reality.

Model Validation Procedures

Rizzi and Vos (1998) outline a procedure for validation, and it is very similar to the goals and intents of the EISMINT (European Ice Sheet Modelling Initiative) experiments. Validation is also a step-wise process, and between the steps of increasing code fidelity lie opportunities for code and calculation verification. In the CFD world, as in the glacier modelling community, the validation process is carried out by comparing representative simulations with trustworthy detailed experimental measurements, and this is done with increasing maturity of the PDE model and its numerical code. At the development stage, research codes that are assumed to accurately model the continuum PDEs are validated by comparison with benchmark experimental test cases of relatively simple flows, on simple geometric configurations, and include a single dominant flow feature. These experiment test cases are wind tunnel or flight simulation experiments for aerospace CFD, but in glaciology the benchmark cases are derived computational models with pedigree in extensive laboratory and field data or in previously verified simple flow models.

At the second stage, the physics and code is extended to include at least two flow features or phenomena that interact with each other. This analysis is done in the CFD case over a component of an aerodynamic system (such as interacting shock wave and boundary layer over a wing). In the glacier modelling case, this stage is similar to the pre-Level II EISMINT model intercomparison experiments that contributed to the 1997 Grindlewald Meeting. At that phase, several aspects of ice-sheet model phenomena that were not addressed in the EISMINT-1 needed exploration, but still over simplified geometry. These aspects included:

1. full coupling of temperature evolution with flow through T-dependent ice rheology;
2. temperature and form response times to step changes in the boundary conditions;
3. divide migration rates in response to accumulation;
4. response to simple, T-dependent sliding laws; and
5. response to topographic variation.

(see Payne: http://www.soton.ac.uk/~ajp2/eismint2/eismint2.htm).
Like the CFD case, these experiments also represent two interacting flow phenomena and are still carried out over simplified geometry.

Finally, the mature stage of validation, one that is anticipated in later EISMINT model intercomparison exercises, is detailed by Rizzi and Vos (1998) as using full production code, relying on comparison with complete systems configuration, focussed on global performance data of the computational model. In the EISMINT case, this would match numerical models against derived Greenland and Antarctic representations and geometries. The test cases for the mature code use full system models not just components, and they involve complex flows with multiple interacting features that represent a specific fidelity and scale of the physics.

To carry out these validation efforts, a degree of model calibration and tuning is required. Although calibration is intended to tune the numerical code with a particular fluid dynamics model in order to improve its predictive capacity of globally integrated (and measurable) quantities, calibration in fact leads to a loss of generality in the model. Hence, calibration must be carried out over several test cases that cover both presence and absence of the various physical phenomena of interest in order to bound the effects. This defines a restricted class of flow problems and features to which the model and code is sensitive, and is hence part of validation.

Code Verification Procedures

At this point, it becomes necessary to intertwine verification steps with the model validation process, for as we have seen, verification of a code for one problem does not guarantee the same level of accuracy for other problems. Similarly, the simulation is only valid for a certain class of problems in that agreement of a calibrated model with reality may be only fortuitous when it is used under conditions different from the original calibration set. Grid convergence testing probably needs to be carried out whenever the problem statement is substantially enhanced to establish or maintain equivalent solution accuracy. This process is almost never done due to the expense of either setting up multiple new grids and running the code while monitoring errors, or running multiple variations of the code across the same grid. However, each instance of verification establishes the level of accuracy and sensitivity of solution results and parameters in the numerical scheme. If the code is new and increasingly complicated, and there is limited experience with it, then it is worth re-verifying that it numerically converges to previously verified versions of code, or to available, manufactured exact solutions.

The sensitivity of the simulation to discretization error is established through convergence to known solutions or through grid refinement studies; for example, by comparing several simulations of the same problem across different grid resolutions. The iterative process is: evaluation of model uncertainty using sensitivity analysis; then validation via comparison with measurements or test case models; next code verification; then extensions of the model; and back to rigorous verification of the new set of equations. As one progresses, it is important to demonstrate the degree to which the previously calibrated models and their uncertainties are transferable to the new problem of interest. Below are two methods for verification of codes: one using recovery of a manufactured analytic solution that exercises all derivatives in the code, and one that bounds grid resolution errors through refinement studies.

Oberkampf and Blottner (1998) describe a method for verifying a numerical code using analytic solutions. Their discussion is a summary of an extensive paper by Steinberg and Roache (1985) that uses MACSYMA symbolic manipulation for the analytic expansion of derivatives as needed in this procedure. Roache (1994, 1997, 1998) further extends and summarizes the method as part of uncertainty analysis for CFD codes, and he includes grid refinement as part of the method. The appeal of this procedure arises when one understands that to verify a code, one does not have to be evaluating convergence to a physically realistic solution. One only has to be sure that all derivatives in the code are exercised, including cross-derivative terms, and that all iterations go to full convergence.

If one has an analytic solution to a PDE, then showing that the numerical model converges to this solution constitutes a verification of that code. Generally, in CFD and in glacier modelling, analytic solutions are not available for the PDEs of interest. Thus, the Steinberg and Roache process is to construct one. First, they choose a specific form of a solution function, similar to the class of problems treated by the code, and assume it satisfies the PDE. Inserting this solution form into the PDE requires expansion of all the derivatives analytically. Simplify the equation, and because the assumed solution does not exactly satisfy the PDE, there will be inequalities of terms between each side of the equation. Now group all terms resulting from this inequality into a forcing function term, Q(x, y, z, t), and add this to the original assumed solution as a source term. This new solution will now satisfy the PDE exactly and is an analytic, albeit not necessarily physically realistic, solution. Next the BCs for this PDE experiment can either be Dirichlet conditions, specifying the new solution function on the boundary, or they can be Neumann or mixed conditions derived from the new solution function. The calculated source term Q and the new BCs must now be programmed into the numerical scheme. The numerical code is then run to convergence, and the converged solution is mapped against the manufactured analytic solution for agreement. If the agreement is acceptable, then a substantial number of numerical aspects of the code have been verified. These aspects include the numerical method and the spatial differencing scheme; the spatial transformation technique used in grid generation; the coordinate transformation used for exercising all terms in the code (discussed below); the grid spacing technique; and the coding correctness and the general solution procedure.

Roache (1998) extends this method to include a grid refinement study, and in doing so also verifies the order of the observed discretization (possibly different from the theoretical order anticipated). The grid refinement study is described below; it shows that systematic truncation error convergence is monitored during several runs of the code over progressively refined grids. Thorough iterative convergence is required. As a metric, the p-th order of error, $E_p = \text{error} \;/\; \Delta^p$, for grid spacing Δ, should remain constant during grid refinement if the code is doing what is expected of it. No drift in this error during refinement verifies that the numerical method is accurate to order p over all points for all derivatives.

One issue that arises is guaranteeing that the chosen solution function exercises all derivatives in the numerical experiment, and generally a coordinate transformation is chosen to ensure this fact. Steinberg and Roache (1985, p. 274-277) select the function as follows. Assign coordinates $x_i = (x_1, x_2, x_3)$ to be

$$x_i = \xi_s + \xi_i + \tanh(d_i\, \xi_1\, \xi_2\, \xi_3) \;, \qquad i = 1, 2, 3$$

where $\xi_i = (\xi_1, \xi_2, \xi_3)$, with ξ_i values being linear from 0 to 1 according to $\xi_1 = h_1 *$ $(i - 1)$, etc. Here h is the scale Δ in the original finite difference approximation. ξ_s represents a zero point shift in coordinate to avoid singularities at the origin. The d_i is a control parameter that adjusts the severity of coordinate stretching. If $d_i = 0$, then there is no stretching in x_i . For non-zero d_i , the tanh function allows non-zero values for all derivatives in the PDE. Steinberg and Roache (1985) note that errors will show up more readily when all coefficients of the PDE operator are of the same order. If this is not the case, it is sufficient to scale the coefficients to help identify the source of the truncation error during grid refinements. Later this solution can be used as guidance for problems wherein the operator coefficients are of disparate size.

Grid Refinement Procedures

Inadequate grid resolution can be a major source of overall numerical solution error. For finite difference schemes, the spatial discretization error can be estimated using the Richardson iterated extrapolation method. The method requires at least two numerical solutions with different grid sizes for the discretization error to be estimated. Usually, the fine grid solution is calculated over double the number of grid points in each direction of the coarse grid. Roache (1994) developed a grid convergence index based on the Richardson method that basically ratios an error estimate obtained on grids that are not doubles (or halves) of each other, and converts the estimate to an equivalent grid doubling estimate. Richardson's iterated extrapolation, or deferred approach to the limit, (see for example Oberkampf and Blottner, 1998, or Roache, 1998) says that for series solutions

$$f_{exact} = f_{discrete} + \alpha\Delta^{p} + \text{H.O.T in } \Delta$$

where p is the assumed-known order of accuracy of the numerical scheme. The coefficient α is a constant for a given solution and does not depend on any particular discretization. Its value is also derived in the refinement process. Two numerical solutions over different grids are computed, and these are combined to compute a new estimate on the exact solution and a measure of the discretization error. Oberkampf and Blottner (1998) point out that in practice, more than two refined solutions will be required. First, the global accuracy of the method over solutions and integrated or differentiated solution functionals (like integrated discharge, or differentiated velocities yielding strain-rate fields) can be degraded due to inaccuracies or errors in implementation. These errors may not show up in the first refinement or may have cancelled out until the refinement shifts resolution scale. Second, the higher-order terms in the extrapolation are not negligible at first because insufficient grid resolution was undoubtedly used on the first few solution attempts. Refinements must continue until the computed grid convergence rate matches the known order of accuracy of the code. At that point, the method can be used to estimate the error between f_{exact} and the discrete fine-grid solution (Roache, 1998).

Richardson extrapolation is best used not to obtain the correct discrete solution, but to obtain an estimate of error by differencing the solutions derived. Accurate application of the method for error estimation, with known order of accuracy p, requires that the observed convergence rate equals the formal convergence rate. When this happens, the leading order truncation error term in the error series is known to dominate the error, so the estimate is accurate. Note, however, that nonlinear features of the equations may contaminate grid convergence, so if excessive refinement seems required to match convergence rates, it may indicate other problems.

If the goal is to verify the actual order of accuracy for a problem rather than using a known p to estimate error, then a variation of this method suffices. The actual observed order may be different from the predicted order based on the scheme because the observed order of convergence depends on achieving the asymptotic range of the solution at small residuals. The observed order may also be different from the order previously verified in a test case. Roache (1998) shows that observed order p can be extracted from operations with three grid-refined solutions, and this p can be compared to the assumed theoretical order. Let f_1 represent a fine-grid solution calculated over grid spacing h_1 , and let f_3 represent a coarse grid solution over spacing h_3 , with f_2 and h_2 being intermediate to these. Let r be the grid refinement ratio defined as $r = (h_2 / h_1) > 1$ or $r = (h_3 / h_2) > 1$, assumed constant but not necessarily $r = 2$. Then from combining Richardson extrapolation equations for each solution, there arises the relation:

$$p = \ln \{(f_3 - f_2) / (f_2 - f_1)\} / \ln (r)$$

where p is the observed order. If r is not constant over these grid sets, then a more general equation must be solved for p. Let ε_{23} and ε_{12} represent the solution differences (f_3-f_2) and (f_2-f_1) respectively, and let r_{23} and r_{12} be the respective grid refinement ratios. Then the general equation to be solved for p is:

$$\varepsilon_{23} / (r_{23}^p - 1) = r_{12}^p \{ \varepsilon_{12} / (r_{12}^p - 1) \}$$

For r not constant, this equation is transcendental in p and Newton-Raphson techniques (or similar) can be used to extract the order p.

One final point of interest in obtaining estimates for discretization error that can lead to methods of examining error propagation. Instead of examining leading order terms in the truncation error, one can approach the problem through spectral methods. An overall solution discretization error says nothing about what grid locations and clusterings, or what parts of the solution, are contributing most to the errors. For series solutions, an energy spectrum of the solution and solution error can be created. These include solutions that can be decomposed into harmonic components, that can be represented by recursion relations, by discrete wavelets, or represented by non-periodic spectral methods such as Maximum Entropy Methods. For localizing an error estimate, short data records must be used, and the Maximum Entropy Methods (Ulrych, 1972;

McDonough, 1974; Press and others, 1992) are especially useful in assessing the frequency spectra of short data records. The method is simply autoregressive spectral analysis carried out in the space dimensions rather than in time. The spectrum indicates the amount of energy partitioned in solution wavenumber (inverse wavelength). The local error associated with every spatial discretization scheme can be modelled as the inner product of the accuracy at that location times the energy spectrum of the solution at that location. If the gridding and analysis is done so that regions of interest can be isolated and local solution results can be extracted there, then one can examine how the energy in the solution falls off at higher frequencies. The total local error in the discrete approximation is then the integral over all wavenumbers of the spatial discretization scheme's error $e_\delta(\omega)$ at location δ times the energy distribution E of the discrete solution $f_d(\omega)$ as:

$$\text{Local Error} = \int e_\delta(\omega)\, E(f_d(\omega))\, d\omega$$

This gives the size of the error where the solution shows the most spectral energy. **Figure 1** is a schematic representation of this situation, and shows graphically the discretization error from this integral. The plot is solution energy E vs. wavenumber ω. The normal growth of error is from low frequency (small wavenumber or long spectral wavelength) into the high frequencies. If the energy falls off in a well-behaved manner as in curve 'a', then there is very little intersection of the solution energy curve with the error growth. However, if the solution energy falls off more gradually as in curve 'b', then high frequencies are being polluted, and there is much more discretization error ('d.e.') in the solution. By evaluating the solution quality and accuracy in small block regions decomposed from original multi-block grids, the contribution of discretization error in those regions can be evaluated independently. If the source of error is upstream of a feature of interest in the model, then this error stands a high chance of propagating or convecting into that region and being enhanced. Formal localization of error sources and growth thus allows exploration of how the solution is evolving, and how the spatial resolution may need to be altered to capture the physics of interest. It may be possible to estimate downstream propagation of error without formally solving an error equation or an error convection equation.

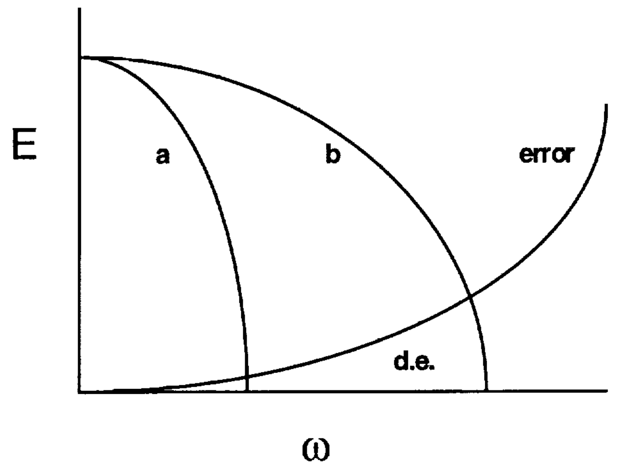

Figure 1. Schematic representation of solution energy **E** vs. wave number ω

SOLUTION QUALITY IN GLACIER MODELLING

Solution quality means solution accuracy. To control solution quality, one must explore how to control or assure solution accuracy. In this section, a series of recent publications in the glaciological literature are examined not for their results, but for their methods. These representative example methods are being used to successfully maintain solution quality, and these papers are offered as models for quality benchmarking, model calibration, validation studies, verification studies, or error analysis.

The glaciology community has recently published a variety of results from extensive numerical experimentation and benchmarking of models and codes for the ice-sheet equation as part of the European Ice Sheet Modelling Initiative (EISMINT) (Huybrechts and others, 1996). Fifteen ice-sheet models were submitted to the model intercomparison tests at the Level I exercise. The exercise published the numerical grid and model constants and parameters on which the participants were to exercise their respective codes. Boundary conditions were established for both a fixed margin experiment and a moving margin experiment. Both steady-state and time-dependent behavior was evaluated, and simulations were to be run over an evolution period of 200 000 years. The submitted models had different ways of calculating the ice fluxes, and the discretization schemes were different; although two broad schemes were recognized. All teams adopted a staggered gridding scheme due to the known problem of unstable performance of non-staggered grids in representing diffusion effects in the flux calculations. An exact analytic solution was available for the 2-dimensional experiments, and thus the various broad categories of schemes could be analyzed for estimates of truncation error in the solution, and for discretization error forced by the grid/mesh interval. These results can provide guidance for accuracy in the 3-dimensional models. A consensus was reached concerning resulting flow and temperature fields for each of the variety of experiments. This consensus provides reference solutions against which future modelling codes can be assessed for accuracy and consistency. Note that the code verification and accuracy was left to each modelling group, apart from comparison of results to the 2-dimensional analytic solution and the consensus achieved through the experiments. Any large divergence in results from the consensus was considered to be caused by numerical inaccuracies. Running the experiments under fixed and moving margins with steady and sinusoidal climate boundary conditions provides a calibration for the participating models. Model validation is not an explicit issue in these experiments because at Level I, the physics is constrained to be rather simple, and only the numerical stability and accuracy of the participating models is sought. Error analysis was presented as variations between model results because most errors were controlled by the details of the experiments.

A second phase of the EISMINT experiments was also run (Payne: http://www.soton.ac.uk/~ajp2/eismint2/eismint2.htm), as discussed above. It was recognized that several aspects of ice-sheet model phenomena had not been addressed in the EISMINT-1 experiments, and additional trials would enhance the calibration and performance of the submitted models. These aspects included sets of two interacting flow phenomena that could be explored over the same planar geometry of the original experiments. In addition to adding calibration and better model validation, these experiments help spawn new papers that used the EISMINT process as the source for benchmark solutions against which to expand physical understanding.

Hindmarsh and Payne (1996) represent such a study in extrapolation of methods. They examine three different discretization schemes for the ice-sheet equation, and run numerical experiments with these using different time-step schemes (marching and non-linear iterations) to isolate the stability features of the methods. Accuracy is determined over various grid resolutions, and time-step limit bounds are developed for the schemes to maintain accuracy. Comparison is made for the accuracy of the various discretization schemes with available analytic solutions in 1-dimension for flow law parameter $n = 3$, and in 2-dimensions with $n = 1$. This provides full verification of numerical accuracy for models of that complexity. Computational efficiency of various iterated and non-iterated time-marching schemes is compared. EISMINT 2-dimensional, $n = 3$, results are used for carrying out extrapolation to more complex models. The use of iterated maps as the representation of the numerical solution is introduced and developed here, and these aid in understanding and controlling uncertainty in model evolution. The onset of numerical instability is found to depend on the method of time-stepping, and a correction vector is developed that represents the evolution of the equations in such a way that appropriately small time steps can be selected without computational investment in extensive specific experimentation. A great part of the value of this paper is the way it specifically controls the accuracy of the numerical experimentation, isolates causes of variability in solutions, and compares several schemes in exploring solution quality.

Marshall and Clarke (1996) use a conventional 3-dimensional finite-difference model, and by employing coupling terms to model mass exchange, they examine sheet-ice and stream-ice components in the same model without having to explicitly develop ice stream physics. Yet, ice stream physics can be explored by these methods. Ice streams are sub-grid at the current resolutions of gridding schemes for the ice-sheet equations. The authors thus convert a complicated physics problem into a form that can be modelled using well-proven codes. They parameterize the creep exchange process between sheet ice and stream ice, and they separately model stream ice fluxes by subglacial bed deformation and de-coupled sliding at the ice-sediment interface. The areal activation of ice streams is controlled by basal conditions. Sensitivity tests are run on EISMINT benchmark configurations, and resulting thickness and velocity profiles are derived and compared. With that heritage, the results enhance the benchmark cases, and the sensitivity analysis bounds the model uncertainty. The mixture of steam ice and sheet ice within the same model, without the need to specifically model sub-grid physics, provides an excellent enhancement to existing models and their use. The validation and calibration of their models are ensured because of the detailed development of the physics and the PDEs before application of the numerical scheme. Assumptions and constraints are explicitly identified, and the ramifications of such simplifications are discussed providing conditions of applicability for their models. Fairly simple mass balance equations can be used numerically because of the extensive work in developing mixture and exchange relations that define the problem statement. The dynamic coupling of the ice mixture is parameterized in such as way as to allow examination of effective control, activation, and development of ice streams even in a full 3-dimensional flow field. Finally, the exploration contained in the sensitivity tests demonstrates the range of applicability of these methods for further enhancements.

Hindmarsh (1997) explores the use of normal modes of eigenvalue problems, derived from linearized versions of the ice-sheet diffusion equation, to initialize models and examine small scale changes in features and response-times. In non-linear models, accumulation rates and viscous properties must be parameterized and tuned in order to calculate ice thicknesses. Such tuning is not needed in linear models, and fluxes are derived directly from balance relations. With the increasing availability of highly accurate digital elevation maps from satellite altimetry, actual ice sheet geometry can be modelled rather than calculated. Here, model validation arises from initializing the numerical models using actual data, and carrying out perturbations about EISMINT solutions. The normal mode solutions are compared to similar normal mode analyses derived from Antarctic digital elevation models, and balance flux results are computed across varying grid scales. The normal mode analysis permits resolution of small scale features in Antarctic elevation models; in non-linear models, the small scale structure in such data is relaxed out due to numerical instability. The linear methods, although having some identified shortcomings, allow modelling and examination of small scale effects. Code verification becomes essentially unnecessary because with the linearization scheme Hindmarsh uses, grid-centered difference fluxes are computed via linear equations and matrix inversions that do not require the verification methods of PDEs. These results also extend the applicability of the EISMINT results, and can be used as additional benchmark test cases.

The EISMINT benchmark test cases and their derived extensions work well for models that assume ice-sheet configurations and aspect ratios. However, when one intends to explore the behavior of valley glaciers or of ice sheets under conditions where the approximations in the simple slab model break down, such as near the ice divide or in ice streams, then new methods must be sought. Some of these conditions were explored in the Marshall and Clarke (1996) sensitivity studies on ice streams. Additional methods are now considered. Models of stress and velocity fields in glaciers are analyzed in two companion papers, one that examines finite difference schemes for higher-order glacier models (Colinge and Blatter, 1998), and one that explores sliding and basal stress distributions using the first model (Blatter and others, 1998). In both papers, substantial headway is made in applying numerical methods to improve grid resolution, and then enhancing the representation of basal velocity and shear models with moving stress concentrations within this framework.

Colinge and Blatter (1998) create several numerical schemes and evaluate their stability for modelling 2-dimensional stress and velocity fields, including stress gradients, in glaciers. In particular, the paper looks at the modelling of conditions wherein the usual approximations of shallow-ice aspect ratio break down. A scaling scheme allows the development of a linear perturbation method on the governing equations. The equations are separated according to order of the small scaling parameter, and numerical methods are examined on these sets of equations. Distinction is drawn through numerical experimentation on the efficiency, accuracy and conditions of applicability for the methods. The equations are transformed so they can be written as a series of coupled, first-order ordinary differential equations and one algebraic constraint. This allows application of the method of lines wherein discretization occurs in all directions except one, and a shooting integration scheme is developed from the glacier bed to the surface. The shooting requires iteration and correction. Because of the kind of coupling in the governing equations, it is found that even for each discretized derivative of order p, the over scheme is of order p - 1. Thus, the derivatives in the algebraic constraint must be discretized to order p + 1 in order to maintain and overall scheme order of p. Boundary conditions at the base are set to be either no-slip or functionally related sliding velocity and shear traction, with a tangency constraint between horizontal and vertical velocities at the base. Experiments are run on each condition. Single shooting schemes are run for both fixed-point iterations and non-linear Newton iterations, and the flexibility and fidelity of each is compared. Additional constraints are discovered between the form of the surface tractions as a function of the respective basal tractions, and this function must be infinitely differentiable to guarantee solution uniqueness. For fixed-point methods, a criterion is developed to ensure existence and uniqueness of the solution as well. A condition number is derived for the Newton iteration method, for both single- and multiple-shooting schemes, which dictates the instability of the algorithm in situations when there is high sensitivity to the initial values at the glacier bed. These methods are applied to mixed basal boundary conditions, prescribing both basal velocities and basal shear tractions over regions of the bed. Experiments with the method show the rise of numerical instabilities and even-odd oscillations over two grid-cells. Through refined exploration, a non-symmetric discretization scheme is discovered that removes oscillations while maintaining well-defined order. The schemes are calibrated and validated for the particular model of parallel-sided plane slab flow, and first- and second-order solutions are derived and compared across a variety of grid point schemes.

Because the idea of shooting schemes is to learn corrections to the initial starting conditions, this method is ripe for sensitivity analysis experimentation that controls instability. Verification is done here by examining the stability of the method and convergence rates, and by evaluating stability criteria that dictate step size. An unrealistic solution arises in which second-order solution profiles show negative shear stress in the interior of the sliding area (p.454), and the authors speculate that this is related to small oscillations around zero-stress values over the order of two grid-cells, and believed it to be numerically induced. Considering the degree of customization that has been developed in this paper for specific discretization schemes, it might be worthwhile to explore manufacturing an analytic solution against which to test the authors' numerical schemes, as advocated by the previously described methods of Steinberg and Roache (1985) and Roache (1994, 1998). The authors indicate that the most serious limitation to their methods (for validation) is the lack of general knowledge of spatial variations of basal velocities and coupled longitudinal stresses at the bed. These conditions are of course the initial integration conditions of the model, and sensitivity of results in these conditions has been previously mentioned.

In an effort to experiment on sensitivity of basal conditions, the companion paper of Blatter and others (1998) uses the schemes developed in Colinge and Blatter (1998), thereby becoming a validation exercise for the previous models and schemes. Here, numerical experiments are carried out to study the interaction between basal velocities and the spatial distribution of shear stress distributions, at several scales. These experiments yield the important result that sliding is not a local phenomenon in cause. Experimenting with both sliding and mixed basal conditions, the authors show that for spatially periodic sliding / non-sliding conditions, a distance of 5-10 times the ice thickness is necessary before the average sliding velocity can be considered uncoupled between one period and the next. Sliding areas are discovered to be responding to conditions that are not local to the observed sliding area, and are in fact responding to conditions within distances on the order of the width or substantial parts of the glacier length. Hence, the theory has provided important insight for interpretation of field data, and the necessity of multiple taps to the glacier bed in order to understand local physics. Multiple numerical experiments are run. Because the average basal shear over the entire bed remains invariant under changes in sliding patterns, this becomes a benchmark metric. If there is local reduction in basal shear traction, then the ice is sliding or the bed is deforming. Calibration of the models is carried out by computing the stress and velocity fields over a 2-dimensional geometry section of the Haut Glacier d'Arolla and discovering that the flow and stress patterns over realistic geometry are the same as for simpler slab geometry. Exact location of sliding cannot be predicted, only that it has occurred. But this calibration maps and establishes a range of applicability of the model. Additional experiments are run under cases of infinite effective width, such as with ice streams that are wide compared to their thickness, or that flow within their own ice channels. These experiments help to isolate the effects of side drag compared to basal drag. The series of experiments in this paper challenge the previously held validity of using flow laws to indirectly determine basal stress components. Even given measurements on basal strain rates, this coupled with knowledge of the flow law does not approximate the basal drag well because of non-locality, and hence these should not be used to derive basal shear tractions. Small spatial scale variations can lead to rapid stress variations, and perhaps migrations of these along large stress gradients. The suite of numerical experiments and sensitivity investigations provides credible validation and calibration of the physical models being proposed. The matching of the derived results against patterns in actual valley glacier geometries lends credibility toward verification of the extensive numerical codes developed in Colinge and Blatter (1998), even though this by itself does not represent a formal verification process.

All of the above papers present excellent strategies for their analysis methods and explanations of the detailed physics captured in the problem. Where possible, codes are verified through comparison with benchmark results or analytic solutions. Where formal verification has not been possible, extensive sensitivity studies have worked to demonstrate the range of applicability of the numerical methods and the model parameterizations. Formal error analysis and explicit ramifications of assumptions have been frequently included. The sophisticated results justify the effort expended to maintain solution quality and to yield understanding of complex flow physics in the glacier environment.

As analysis becomes even more complicated and data-sets become more diverse and heterogeneous, it becomes increasingly important that the codes and methods be controlled through a managed process of experimentation. Validation and verification by teams of researchers, each with access to others' results and methods, will become more needed in order to control the expansion of information and refinements in flow physics. In the following section, an information architecture is presented that will help unify and manage the physics explorations, will help deploy field and satellite data in service of boundary conditions. The system will also keep track of the physical problems and limitations for which particular codes and discretization schemes are applicable.

THE ICE COMPUTATIONAL ENVIRONMENT

An Integrated Cryospheric Exploration (ICE) Environment is now proposed for exploring methods and managing sources of uncertainty in glacier modelling. The details and functionality of this Environment are described based on modifications of a system developed during the past two years at NASA Ames to support aerospace Computational Fluid Dynamics analysis. The original system was created in concert with aerospace companies who were using wind tunnels for testing and validating aircraft wing and wing element re-designs for purposes of improving the flight performance characteristics of the original design. First, wind tunnel test results were made available on-line during a single test entry. Then, multiple test entries were archived and compared. Next, there was a desire to have CFD results across the same configurations made available during the wind tunnel tests, and to have the results of CFD exploration provide suggestions of enhanced design configurations or changes in the model. These changes could then be rapidly manufactured as new physical models for testing. Ames personnel built an interactive infrastructure that launches a variety of CFD flow solver codes across highly complicated wing/element geometries and grids, and organizes the resulting solution fluxes, forces, moments, and integrated parameters into a form that is accessible for visualization and design refinement decisions. The current version of this system is now called the Advanced Design Technologies Testbed (ADTT). The ICE Environment is being developed using the experience from ADTT development, and it consists of solver codes and models inserted into that same information infrastructure. A discussion of ADTT functionalities follows.

Description of the ADTT

The ADTT supports multiple functions for grid generation, initializing solver codes, running and monitoring the codes, and analyzing the CFD solutions. A geometry toolkit is available on-line that allows simple grid generation over simple geometries. It also links in full overset grid technology methods that are developed off-line for complex geometries, and it can merge off-line and on-line grid generation for mid-range fidelity geometries and gridding. The resulting grids are then used by multiple flow solvers. Parameterized CAD objects allow for non-expert use. Input files and CAD automation change easily in response to changes to parameters in the user interface. The solver code initialization window prompts users with standard parameter values, but allows flexibility in resetting these. The entire user interface is backed up with dependency relations that maintain feasibility in problem set-up, and thus the user avoids launching a code with parameters mis-identified or with improper values, or with gridding parameters un-matched. All auto-gridding and auto-launching relies on rule-based schemes, but an unlimited number of gridding or solver processes can be organized and launched from a single Graphical User Interface. The jobs are partitioned and directed according to customized encoded rules. This enhances process control and bookkeeping of the numerical experiments and their variations. Furthermore, multiple geometric configurations can be set-up and linked in a series of runs in order to explore variations in a particular design parameter space. Such variations can be set-up as special cases, but it is not done automatically and relies on the user to create, maintain, and deploy the overall strategy.

Various finite difference schemes are supported and can be changed out, but none of this selection is automated. All differencing schemes are set-up off-line, then linked in and launched according to the numerical experiment to be run. However, multiple schemes can be used and compared. Most boundary conditions are encoded with applicability relations that constrain their use to particular problems so as to avoid non-physical flow results. The turbulence models and other auxiliary equations are likewise encoded with constraints, some of which are suggestions only, whereas others cannot be overridden and are locked in as a default setting in the codes.

A variety of flow solver codes are supported, each with increasing fidelity and complexity. They are Fortran codes that were written before the ADTT was created, so they are wrapped and linked in by Unix scripts, and launched on a multiple CPU SGI cluster. All codes are initialized and launched from the User Interface that maintains dependency relations between code parameters. There are two ways the codes are monitored during a run. First, the user can pre-set a threshold parameter or target convergence residual in advance (interim or final), and the code will launch and run to the threshold, publish its state, then continue or use up an allocated time resource. Interim check-point files can be saved along with final results in a directory structure if this feature is set-up before the code is launched. Such interim files can also be saved to a database for future analysis of trends and error analysis. Second, a Procedural Reasoning System (PRS), that was developed under external contract, has been linked in that monitors the state of progress of the solver. The PRS takes in information about the executing code and matches that against a library of procedures that can monitor and dynamically adjust the flow solver activities. At the simplest level, the PRS can detect difficulties in convergence, and adjust solver parameters to the experiment during runtime. Multiple procedures can be spawned and execute asynchronously, so one process can be kept running while, for example, the time step might be changed in a parallel process to see if this helps convergence. The PRS can be user-driven or set to operate autonomously after certain defaults are recognized. A generic post-processing strategy is not currently available, but may become useful if the whole system becomes CORBA compliant. Currently, strategies for post-processing and analysis are maintained off-line and can be linked in for a given numerical experiment. Additionally, if a strategy were to deploy a variety of boundary conditions to test sensitivity, this could be organized at the beginning of multiple runs, but the current system does not allow interruption at such a level during a run. Such a change would have to be deployed asynchronously to the original process, but it could also be monitored by the PRS.

ADTT has been able to function well using commercial visualization packages and tools. Retrieval from the database is based on query or tree-browser object/data attribute selection. Two-dimensional data plots, flow field, vortex, or streamline representations, and contour plotting, with web-enabled remote sharing of views has turned out fully sufficient. Originally, it was envisioned that customized data presentation packages and plotting routines would be needed, and several advanced visualization systems have been created under separate funding that can respond to the data streams generated from traditional CFD output files or from wind tunnel test data. However, to date, such customized routines have not been necessary within ADTT since most visual analysis for the CFD is done by plotting routines, wherein the user picks contour intervals, boundaries, and database variables and these are brought up to a plot window by Java applets or simple search agents. Feature recognition capabilities exist, but this is dependent on the code resolution and is hence problem and solver code dependent. The only error recognition currently used is based on convergence criteria. This is generally because the user interface has guaranteed that the problem launched is feasible, and if there are global problems in solution, they will show up in non-convergence. At that point a separate verification activity would have to be undertaken. To date, this has been unnecessary in the aerospace design process because the design has been tested first on proven, lower-fidelity codes (2D-incompressible, 2D plus sweep in a third direction), and the full Reynolds averaged Navier-Stokes codes are only used sparingly to refine the results of the multiple, lower-fidelity runs. However, there is no reason that a verification strategy could not be created and deployed across the system. There is currently no automatic linking of interim or post-processing results back to the code initialization window to re-start a current run or set-up a new problem sequence. Once particular methodologies are learned from experience, then rules and models and their dependencies can be extracted that help automate this process.

A customized data management system has been created, but it is not generic. It is a simple directory structure that keeps track of solver runs for a particular problem configuration. Rizzi and Vos (1998) have suggested a hierarchy for CFD databases that represents a cascade structure with each flow regime under study at the top of the hierarchy. Then under each aircraft flow regime, there could be large sets or subsets of systems or aircraft types, followed by integral subsystems and components, and finally features or flow phenomena that occur over these components. Once such features would be recognized, they could be used to prescribe what kind to modelling would be needed to refine the flow regime resolution. In ADTT, there is no sophisticated indexing on context or particular flow features because for the problems currently presented to ADTT from industry, there has always been a very detailed configuration and geometry that is being tested. This geometry drives any hierarchy, so indexing on the geometry is sufficient. Features are sought that occur over the particular configuration of interest, and these are indexed to that geometry. ADTT deploys results to a separate on-line system for comparison of its CFD results with relevant wind tunnel test results, and the results of this comparison by designers starts an effort to create new physical models.

ADTT in practice has intentionally not run any code verification experiments because the codes are very well known by the developers. The projects have not required any experiments on validation because this is done by the test engineers who compare the CFD results with the wind tunnel test benchmark cases. There is no on-line adaptive-gridding or feature-following boundaries in the codes to date because the gridding has been done with the code developers who are part of the industrial trials. For re-design problems, they already know what flow characteristics they are trying to resolve. Questions as to what conditions have been selected, which grids and why, and so on, that might arise during an exploration experiment have not arisen during ADTT project use, because the project is always driven by the design and purpose for the test -- to match and compare results with wind tunnel configuration data. The flow solver code is selected for the problem fidelity of interest, and assumptions are tied back to this problem. There have been experiments that compare code performance, but usually this is done by comparing results between codes of differing complexity to see what physics is captured by each.

Development and Use of the ICE Environment

The ICE Environment is being modelled after the ADTT, and many of its functionalities are identical, though not yet fully completed. ICE is an information architecture that can integrate multiple existing glacier flow models over simple bed geometries with boundary conditions and auxiliary process models, in particular models for basal conditions. These boundary conditions are derived from datasets from glaciers, from climate and precipitation data, or from derived mass balance and basal conditions data that are being encoded in a relational database. The process models that look at spatial discontinuities in stress, velocity, temperature, and water pressure are being coded from published literature and will follow a benchmarking process similar to that proceeding within the EISMINT experiments. Field work is planned for the austral summer on recently deglaciated regions near wet, active glaciers on the west coast Southern Alps of New Zealand. This data will also be included to support basal conditions models. The web-based user interface for ICE constrains problem set-up, launches and unifies these codes and models, and it also compliments the analysis results with visualization tools in which non-customary or alternative relations between archived datasets and model results can be created on-line and explored. Parameter space can be navigated visually and interactively, and stiffness of parametric regions against perturbations or variations can be explored to locate model and process sensitivity and applicability boundaries, and hence regions of computational risk.

The ICE Environment is envisioned for exploring and managing sources of uncertainty in glacier modelling codes and methods, and for supporting scientific numerical exploration and verification. It can also support analysis of model's sensitivity to variation of parameters and processes, for locating changes during convergence, or for comparison of alternative till deformation models coupled to flow models. The Environment is a structure, so it is not constrained to certain fixed-use scenarios. But likewise, the codes and models to be used must be scripted to allow insertion into the structure, launching, manipulation, and monitoring of process flow. The Environment can be used in developing a computational experiment or exploratory methodology for initializing, launching, and comparing codes and results using the supporting visualization tools. It can thereby support validation, calibration, benchmarking, verification, and error analysis. A user would develop a strategy, and this strategy is imposed across the elements of the Environment.

Implementing such strategies and scenarios with flexibility requires access to a database system rather than merely a data directory structure. Rizzi and Vos (1998) recommend a future database structure that starts with a flow taxonomy rather than the geometry-driven structure current in the ADTT. For ICE there needs to be a hierarchical taxonomy of features and structures of flows of interest. The features in the taxonomy are then populated with experimental data, field data, satellite remote sensing data, or data and results from previously validated flow and process models. Then representative benchmark cases could be defined for each flow structure or model type, linking the data for pedigree and index. These benchmark cases can then be used to construct the database system. The system would include data manipulation tools that could generate synthesis plots from across models and heterogeneous data types. A data-mining technology is warranted that accesses this taxonomy and model results, and that presents or displays data for correlation or comparison on demand. If the system is accessible over the Web, then the cases can be used by researchers at their home sites to create data plots, and to validate their codes or linked models. Appended to the cases through metadata structures would be the knowledge of which physical models and auxiliary models give the best predictions for flow behavior. Data exchange standards or full CORBA development would support a unified fidelity in data representation, and this would need to precede implementation of the remote-access system.

The data should be allowed to evolve and be replaced, upgraded, refined, and corrected. But it would still need to be tied to the particular benchmark case or flow structure as its base pedigree. This is accomplished through indexing on metadata. The metadata are descriptors of what is in a particular dataset or results file. Instead of searching or retrieving directly from sets of heterogeneous data, the metadata can be searched and retrieved based on key words, on a more complicated indexing that includes context, or on models of the data structures and relations themselves. Each of these indexing structures can be cross-referenced. Particular information retrieval starts at the metadata level, and relevant content to a particular posed query is reorganized and surfaced. Data used to need to be strictly formatted in advance to be retrieved and presented on demand. This is no longer necessary with current manipulation tools and search agents. The data can be reformatted by an agent so it can be compared and plotted on demand. This includes versions of extrapolation, interpolation, and re-registering methods, but of course such manipulation can introduce errors and bias.

A feature that would be very useful in ICE for exploration of multiple numerical experiments would be to have the data sets and model results managed by a software agent in such a way that all archived data, from whatever source, could be selected and used to initialize new modelling runs. The software agent would pull out values and re-populate the namelist or initializer of the flow solver. The extraction by the search agent would feed a record of actions taken, it would be driven by an overall external or encoded experiment strategy, and so these actions would become a process record of the experiment flow. This feature is not currently part of the ADTT system; but technology is available for such a feature, and ICE will be a good testcase for its development. Thus, the database system and its agents would become an integral part of the experiment flow: strategy initialization, problem set-up and initialization, flow solver launch, post-process and archive results, visualization and analysis (perhaps asynchronously during a run), and finally redefinition and restart of a similar or new problem. Of course, the database system can remain fully functional without being directly tied into the flow solvers or other analysis codes, and it can be run entirely by the users.

An additional use for the ICE will include exploration of what approximations in the related basal process models are sufficient for resolution of a problem or feature, and what is the fidelity and sensitivity of the information. A strategy might be to compare and contrast various types of bed conditions, extracting stress, velocity, and temperature data from current models or data-sets. The goal would be to create proxy representations of the detailed models that would be sufficient for certain flow problems or validation exercises. For example, under what conditions would a boundary layer model suffice over a full basal process model, and for which problems is the full model required versus simply a boundary condition? Are the test cases consistently better or worse than simpler models that use mean quantities?

Once verification of models moves from EISMINT's Level I (Huybrechts and others, 1996), wherein modelled processes and parameters are fixed, to Level II, wherein individual models are run including whatever processes are considered important along with modeller's preferred values of parameters, there is substantial risk of moving out of the range of applicability of various models against the more complicated flow situations being simulated. The ICE Environment will help understand and control that risk through managing consistent numerical experiments, and it will enhance uncertainty management by aiding in recognizing sources of numerical error during convergence histories, or causes of non-physical solutions due to mismatched applicability. The simpler models have analytic solutions for verification. As problems become more complex, analytic solutions will not be available. The more complicated models and linked processes will have to rely on constructed solutions for verification, on comparison with previously verified and validated solutions, and on consensus physics as targets for validation.

The ICE Environment can be used for error analysis experiments both through tracking of features and their changes as solutions evolve, and between alternate representations of solutions. Strategies to explore impacts and propagation of errors must be developed whereby a series of code variations are launched. Since errors start as local features, and may convect downstream in the flow field, a visualization capability that created a "cone" of error flow, and an assumed Gaussian error growth distribution could be modelled within this cone as an estimate for later model interpretation. In addition, such an error model could be compared with results derived from field data that was stored in the database. This experiment would also help establish whether uncertainties associated with calibrated models are transferring (or are even transferable) to other problems of interest. The database agents would use interpolations and extrapolation between archived data sets and model results to follow the error just as if it were a flow feature.

Finally, the ICE Environment allows one to create and explore new physics models and representations, even if they are believed to be ill-founded at the outset. There is a long and successful heritage in the development of the appropriate governing equations for glacier and ice sheet models. They incorporate nonlinear constitutive relations and mathematically tractable assumptions such as the shallow-ice approximation, absence of significant contribution from longitudinal stress gradients, and of course the comparative insignificance of inertial terms in momentum balance due to relative adjustment time scales (Colinge and Blatter, 1998). It is not the Author's intention to argue that glacier flow should not be fundamentally Stokesian. However, the scalings and simplifications that are commonly used and grounded on the observed behavior of glaciers (at least in their quiescent phase) are fundamentally unable to capture transient flow behavior or transition conditions between quasi-steady states. For climatic scale relaxation analysis, transition conditions are irrelevant. But fundamental changes and complicated flow structures seem to be occurring at active glacier beds on rather short time scales. These processes are of interest in and of themselves, but also as part of the dynamic system of the glacier and its responses. Exploration strategies can be developed to begin to evaluate this physics, even if it does not appear mathematically tractable at first.

For example, suppose one wants to model the mechanisms of how the ice flow transitions to its new state. The time-dependent continuity model contains in its heritage the understanding first formally articulated in several papers by Johannesson and others (1989a, 1989b) and Raymond and others (1990) that the response time scale for glaciers and ice sheets is that of the "volume time scale." This time scale is a measure of the time for a glacier or ice sheet to reach a new equilibrium state following a climatic or mass balance event. Thus, computational time steps and non-dimensionalized time are regularly scaled to be of order [H]/[a] in most analysis models since the mid-1980's, whether for kinematic waves or for long term climatic adjustments. This means that the shorter duration transient behavior that is on the scale of the "first awareness" in the glacier that a change is occurring, should perhaps be scaled differently so that rapid, transient flow states may be captured and modelled. Steep, fast, wet glaciers may seldom be in steady state, and longitudinal stress gradients as well as complicated basal processes may contribute significantly to multiple flow states. Hence, certain spatial scaling assumptions based on aspect ratio that ignore stress gradients may break down. The appropriate time scales for these conditions may also be better represented by the short time scale τ_s (Raymond and others, 1990) which is similar to the 1/e folding time response of a system — the time between the climate event and the occurrence of first changes in the system (as opposed to the time to fully relax to a new equilibrium state). Thus, the equations of motion, while still not including inertial terms, might be scaled differently in time from the normal spatial scaling in order to capture such flow physics, and these types of analysis procedures would need to be verified anew. An experiment could be devised to use the method of multiple time scales (Nayfeh, 1973; Kahn, 1990) to explore the ramifications of such scalings, whether physically realistic or not, and see where they break down numerically. This way, the rationale behind the scalings can be teased out and the degree of inflexibility in modelling assumptions can be challenged. The experiments may fail due to errors induced by inappropriate time-steps in the numerical scheme that are meant to guarantee stability. But usually these time-steps are calibrated according to the relaxation scale of the problem being modelled. Numerical experiment with such strategies are not meant to disrupt the main agreements in the glaciology community; rather, they are meant to see how the failings of such experiments are tied back to better understanding of the physics or numerical methods, and thus to better understand uncertainty.

CONCLUSIONS

Extensive analysis of the sources of uncertainty in mathematical modelling and numerical representations relevant to computational fluid dynamics and glacier modelling has been presented. Methods for code verification and model validation that are becoming more frequently used in the aerospace CFD community are presented in this paper in an effort to bridge these methods into the glaciological modelling community. Although substantial differences exist between the physics of glacier and ice sheet flows and that of aerodynamics, the numerical verification methods on the finite difference schemes, boundary conditions, and auxiliary relations can be quite similar. Model validation strategies for PDEs consist of both using benchmark test cases containing analytic solutions or known data results, and using calibration of models to establish refined applicability conditions that constrain the uncertainty inherent in the mathematical representations. Sensitivity analyses are found to be necessary methods for both research communities. Specific analysis of the plausibility and ramifications of both physical and numerical assumptions become not only important for credibility, but such analysis also dictates model interpretation methods.

Several example papers from the recent glaciological literature are presented to demonstrate a suite of validation and verification methods and their applications. Some suggestions are included to further refine these methods; but on the whole, these papers are primarily presented as target examples of excellent verification and validation efforts. The physics of glacier flow and the detailed conditions of the glacier beds are being modelled to ever increasing complexity. As more satellite and field data becomes available in heterogeneous forms to be used in model initialization or results validation, it becomes imperative to carry out numerous numerical experiments in an effort to bootstrap verification and validation methods into this more refined analysis. Error analysis needs to be published along with the computational results. To this end, an information architecture that has been developed for use by CFD researchers and design aerodynamicists is presented and described as a tool for managing such experiments and explorations. It is currently being modified for specific use in glacier modelling, and its content and a variety of scenarios for its use are presented.

ACKNOWLEDGEMENTS

I thank Mary Livingston and Robert Childs for stimulating discussions on various aerospace CFD parts of this paper. I also appreciate the numerous valuable insights and review comments from David Korsmeyer and David Alfano on the overall paper content.

REFERENCES

Blatter, H., G.K.C. Clarke, and J. Colinge. 1998. Stress and velocity fields in glaciers: Part I. Sliding and basal stress distribution. *J. Glaciol.*, **44**(148), 448-456.

Colinge, J. and H. Blatter. 1998. Stress and velocity fields in glaciers: Part II. Finite-diference schemes for higher-order glacier models. *J. Glaciol.*, **44**(148), 457-466.

Hindmarsh, R.C.A. 1997. Use of ice-sheet normal modes for initialization and modelling small changes. *Ann. Glaciol.*, **25**, 85-95.

Hindmarsh, R.C.A. and A.J. Payne. 1996. Time-step limits for stable solutions of the ice-sheet equation. *Ann. Glaciol.*, **23**, 74-85.

Huybrechts, P., T. Payne, and the EISMINT Intercomparison Group. 1996. The EISMINT benchmarks for testing ice-sheet models. *Ann. Glaciol.*, **23**, 1-12.

Johannesson, T., C. Raymond, and E. Waddington. 1989. Time-scale for adjustment of glaciers to changes in mass balance. *J. Glaciol.* **35**(11), 355-369.

Johannesson, T., C.F. Raymond, and E.D. Waddington. 1989. A simple mehtod for determining the response time of glaciers. In: *J. Oerlemans (ed.), Glacier Fluctuations and Climate Change,* Kluwer Academic Publishers, 343-352.

Kahn, P.B. 1990. *Mathematical Methods for Scientists and Engineers.* New York, John Wiley & Sons, Inc.

Marshall, S.J. and G.K.C. Clarke. 1996. Sensitivity tests of coupled ice-sheet / ice-stream dynamics in the EISMINT experimental ice block. *Ann. Glaciol.*, **23**, 336-347.

McDonough, R.N. 1974. Maximum-entropy spatial processing of array data. *Geophysics*, **39**(6), 843-851.

Mehta, U.B. 1991. Some aspects of uncertainty in computational fluid dynamics results. *ASME J. Fluids Eng.,* **113**(12), 538-543.

Mehta, U.B. 1998. Credible computational fluid dynamics simulations. *AIAA Journal*, **36**(5), 665-667.

Nayfeh, A.H. 1973. *Perturbation Methods.* New York, John Wiley & Sons, Inc.

Oberkampf, W.L. and F.G. Blottner. 1998. Issues in computational fluid dynamics code verification and validation. *AIAA Journal*, **36**(5), 687-695.

Oreskes, N., K. Shrader-Frechette and K. Belitz. 1994. Verification, validation, and confirmation of numerical models in the earth sciences. *Science*, **263**(4 February), 641-646.

Press, W.H., S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery. 1992. *Numerical Recipes (Fortran). Second Edition.* New York, Cambridge University Press.

Raymond, C.F., E.D. Waddington, and T. Johannesson. 1990. Changes in glacier length induced by climate changes (Abstract). *Ann. Glaciol.*, **14**, 355.

Rizzi, A. and J Vos. 1998. Toward establishing credibility in computational fluid dynamics simulations. *AIAA Journal*, **36**(5), 668-675.

Roache, P.J. 1994. Perspective: a method for uniforming reporting of grid refinement studies. *Jour. Fluids Eng.*, **116**(3), 405-413.

Roache, P.J. 1997. Quantification of uncertainty in computational fluid dynamics. *Ann. Rev. Fluid Mech.*, **29**, 123-160.

Roache, P.J. 1998. Verification of codes and calculations. *AIAA Journal*, **36**(5), 696-702.

Steinberg, S. and P.J. Roache. 1985. Symbolic manipulation and computational fluid dynamcis. *Jour. Comp. Phys.*, **57**, 251-284.

Ulrych, T.J. 1972. Maximum entropy power spectrum of truncated sinusoids. *Jour. Geophys. Res.*, **77**(8), 1396-1400.