Part II

Cryoaction In Silico

Computers make it easier to do a lot of things, but most of the things they make it easier to do don’t need to be done.

Andy Rooney

A Note on Modeling and Simulation

The wisdom consists in knowing
when to avoid perfection

The Murphy Law Complete

“Modeling” and “Simulation” are frequently used as synonyms. In computational heat transfer (CHT) and computational fluid dynamics (CFD) communities, however, the former usually refers to the development or modification of a model while the latter is reserved for its application [4].[1] The straightforward use of the model is referred to as a *direct* problem. From a practical point of view, the reversed formulation could be more useful: how one should place cryoprobes to provide the destruction of the malignant tissues while keeping healthy organs intact. This *inverse* problem is, unfortunately, ill-conditioned (or not-well-posed) and its solution requires some kind of regularization [992] that frequently is just the restriction on the space of possible solutions.

The necessary stages of the code development are verification (an assessment of the correctness of the model implementation) and validation (an assessment of the adequacy of the model to the real world) [825].

Verification is primarily a mathematical issue [825]. The five major sources of errors in the numerical solution have been listed in [714]: insufficient spatial discretization convergence; insufficient temporal discretization convergence; insufficient convergence of an iterative procedure; computer round-off; and computer programming errors. The errors of the last type are the most difficult to detect and fix when the code executes without an obvious crash yielding “moderately incorrect results” [714]. The study reported in [396] revealed a surprisingly large number of such faults in the tested scientific codes (in total over a hundred both commercial and research codes regularly used by their intended users).

Verification is performed by comparison of the numerical results with “highly accurate” (benchmark) solutions. There are three main sources of

[1] This usage problem is language-dependent: for example, in Russian both terms are translated by the same word “modelirovanie.”

such solutions: (1) exact analytical solutions; (2) benchmark solutions of ordinary differential equations (ODE); and (3) benchmark solutions of partial differential equations (PDE).

The usual use of ODE solution is based on exploiting the symmetry properties: one can solve an essentially one-dimensional problem (e.g., having spherical symmetry) using a general three-dimensional grid.

Additional tests could be constructing via a so-called “method of manufactured solutions” by inserting an arbitrary analytic function into the equations and evaluating for what source terms and boundary conditions this function will be an exact solution. This approach is attractive since it could be applied to complex multidimensional domains while allowing direct assessment of the solution accuracy. However, its usefulness depends on the similarity of the guessed “exact” solution to the typical solution of the real problem in question.

An availability of the analytic exact solution greatly reduces the CPU time needed for testing nonstationary codes: one can use an exact solution for the specification of initial and, if necessary, (nonsteady) boundary conditions and avoid the need of using large time integration intervals.

Validation is the second step in the assessment of the software quality. It should be stressed that verification should not be skipped: successful validation alone does prove the reliability of the code due to limitness of a set of validation cases, a “graphical” comparison with experimental data in the most cases and a possible compensation of multiple errors that can give can an impression of a success.

Sometimes it is stated that nature of the code (research, pilot, and production) is determined by its maturity with respect to the validation level [2]. This seems to be oversimplification. If one needs a single criteria (ideally, quantifiable) to estimate the practical usefulness of simulation, the best choice is probably the reliability of a computer prediction [716].

It should be stressed, however, that this parameter characterizes not the code itself, but the simulation, being depending on the adequacy of the model and the accuracy of the computations as well as on the particular aim of the simulations. Evidently, the same results could be considered successful if one is interested in unveiling some trend, and unsatisfactory if the goal is to find, for example, the exact position of cryoprobes.

The last introductory comment should be made on the model completeness. Evidently, one has to search for a compromise between the model completeness and its tractability [1127]. The right attitude to the model development could be drawn from a letter by Blaise Pascal (14 Dec 1656): Je n’ai fait celle-ci plus longue que parce que je n’ai pas eu le loisir de la faire plus courte.[2] Thus, model development is finished not when all the relevant physical phenomena are accounted for, but when you could not simplify the model without corrupting it. This final model could be referred to as a “minimal model.”

[2] I have made this letter so long only because I did not have the leisure to make it shorter.