




# Reproducibility and the Concept of Numerical Solution

Johannes Lenhard<sup>1,3</sup>  · Uwe Küster<sup>2</sup>

Received: 18 May 2018 / Accepted: 16 January 2019 / Published online: 23 January 2019  
© Springer Nature B.V. 2019

## Abstract

In this paper, we show that reproducibility is a severe problem that concerns simulation models. The reproducibility problem challenges the concept of numerical solution and hence the conception of what a simulation actually does. We provide an expanded picture of simulation that makes visible those steps of simulation modeling that are numerically relevant, but often escape notice in accounts of simulation. Examining these steps and analyzing a number of pertinent examples, we argue that numerical solutions are importantly different from usual mathematical solutions. They do not merely approximate the latter, but introduce new problems, including issues of artificiality, stability, and well-posedness. Consequently, simulation modelling can attain reproducibility only to a certain degree because it is working with numerical solutions (in a sense we specify in the paper).

**Keywords** Artificiality · Ill-posed problems · Mathematical solution · Numerical solution · Reproducibility · Stability · Tractability

## 1 Introduction

Alchemists combined practical skills with curiosity and sometimes audacity. They thought the processes of alchemy depended not solely on the objective side of an experiment but also on the personality of the experimenter. Using a grain of salt, an impure character cannot expect to produce pure gold, no matter how meticulously he followed a certain recipe. From the standpoint of modern science, this view sounds utterly strange. That scientific results are reproducible in a non-subjective sense is a key property of scientific method. On this point, all philosophical accounts

---

✉ Johannes Lenhard  
johannes.lenhard@uni-bielefeld.de

<sup>1</sup> Philosophy of Science and Technology of Computer Simulation, High Performance Computing Center, University of Stuttgart, Nobelstr. 19, 70569 Stuttgart, Germany

<sup>2</sup> Department for Numerics and Libraries, High Performance Computing Center, University of Stuttgart, Nobelstr. 19, 70569 Stuttgart, Germany

<sup>3</sup> Department of Philosophy, Bielefeld University, Universitätsstr. 25, 33615 Bielefeld, Germany

of scientific method agree. In practice, however, it might be difficult to reproduce an experiment successfully. Collins (1985), for instance, pointed out that implicit knowledge on the side of the experimenter might play an important role. Nevertheless, reproducibility remains a core element of philosophical accounts of scientific method. If an experimental result cannot be independently reproduced, it is not accepted as scientific.

No wonder, then, at the shock that propagated through psychology when researchers tried to reproduce results published in prominent venues, but failed in a surprisingly high percentage of cases (Open Science Collaboration 2015). This finding triggered a “crisis of reproducibility” that threatens the scientific status of the discipline. At first view, sciences that rely on formal mathematical models, and especially on computer simulation models, seem to be immune against this threat. They possess an almost logical guarantee: Though a model might be highly idealized, or irrelevant, or even completely wrong—still its results will be perfectly reproducible.  $A = A$ —just repeat the entire calculation, or the entire run on the computer. The crisis of reproducibility might affect sciences that *pretend* to rely on mathematical models, but in fact do not. Maybe those sciences include tacit and dubious additional practices that prevent the results from being reproduced.<sup>1</sup> In short, the problem of reproducibility lies outside the realm of simulation models.

In this paper, we argue that the above statement is erroneous. Reproducibility is a problem that concerns simulation models. Moreover, it is a particularly interesting problem because it involves the very concept of numerical solution. The reproducibility problem gives rise to thinking about the concept of numerical solution and about what simulations achieve in terms of numerical solutions.

In fact, the simulation community is aware that ensuring reproducibility requires special measures. Investigating such measures is a vibrant field of research on several fronts, like the interchangeability of different numerical methods or how compilers should work so that they mutually reproduce their results. Research of this kind is about avoiding a crisis of reproducibility. In practice, however, simulation modelers do not restrict themselves to the secured methods, but try to tackle complex tasks with those means that are available to them.

Several indicators from very different directions point toward a potential crisis of reproducibility in simulation-based sciences. Here is a sample: Kaminski et al. (2018) discuss reproducibility in the context of epistemic opacity. They refer to Ludwig (2017) who reported significantly different flow patterns in different computational jobs from a fluid dynamics simulation that were supposed to deliver identical results. Schappals et al. (2017) present a round robin study where a standard (mathematical) model in molecular dynamics produced different results when run by different (experienced) working groups. Reproducing the model’s results by different groups proved difficult for a number of reasons. We will touch upon some of them later in our text. Their study complements the one of Lejaeghere et al. (2016) who

---

<sup>1</sup> There is a less friendly interpretation of the crisis in psychology that says results cannot be reproduced because the publications include outright fraud. No formal models of any kind can be an effective remedy then.

got a positive result about the (statistical) consistency of results of different simulation models that implemented density functional theory (DFT). The positive result, however, should not occlude the insight that the question was open and critical at the start of the investigation.

Additionally, there are informal observations from the supercomputing center where one of the authors has led the numerics and libraries group. Among other things, this department supports clients from inside and outside of academia who plan to use high speed computing facilities for running their models. Quite regularly, clients expect that results are perfectly reproducible and complain when different runs obtain different results. In their opinion, this testifies something went seriously wrong. Simulations with identical initial conditions should bit-wise reproduce one another, so the clients reason.<sup>2</sup> Reproducibility in this sense is a common test (on serial machines) whether a simulation is valid.

From the 1990s onwards, computing centers utilized parallel architectures, but parallelization in a way counteracts reproducibility. Contrary to the naive expectation, two runs will in general not produce the same result. What is reproducible (repeatable) on serial machines is not reproducible on parallel architectures because exact repetition in the order of steps is not guaranteed after parallelization. The same limitation may apply even on serial machines after compiling the program by different compilers or with different options of the same compiler. The basic point is simple: Adding a list of floating point numbers will yield a result. Since machines have limited representation capacity per number, in general this result will not be perfectly exact. While adding the same list of numbers in the same order again will give the same result, floating point addition of more than two numbers is not strictly associative due to the round-off errors. Expressed in simple algebra, it can happen that. The equation holds only approximatively. For sums with many elements as they appear in large linear problems this might result in large differences in dependence on the order of operations. This is not big news, but is reflected in standardization of Floating Point Arithmetic by the IEEE-754 Standard. Hence adding the same floating point numbers in a different order will in general not give the same result as before. And parallelization typically drops the fixed order in which computations are executed. The order might also be changed by different compiler optimization options or by using different hardware features of a processor core as SIMD units. Consequently, bit-wise reproduction is not adequate as a criterion for validation. Admittedly, there are attempts to ensure this type of reproducibility by synchronizing the parallel steps in a defined order. But this is a very sensitive operation and degrades the performance of the program on the parallel machine. People working on current high performance computers have learned that the goal of bit-wise identity is misleading. Bitwise reproducibility on this level is not attainable.

There are also practical limitations. Often, supercomputing power is used where goals cannot be reached on smaller machines. Clients want to use the maximum computational power and the maximum time slot available to get a result. This

---

<sup>2</sup> If reproducibility (across different groups and machines) is discerned from repeatability (same group, same machine), then the clients expect repeatability.

rationale renders reproduction time-consuming and expensive. A very similar observation can be made about clinical trials of pharmaceuticals. Sometimes they are so expensive that they can hardly be reproduced (Ioannidis 2005, see Solomon 2011 for a philosophical overview).

These observations indicate that reproducibility is a problem for simulation modelling practice and philosophy. Of course, we mentioned already that there are thriving research agendas in numerical error analysis and interchangeability of numerical methods whose results give insight into conditions and level of reproducibility (cf. Fillion 2017; Fillion and Corless 2014). However, we target a wider array of factors that influence reproducibility in existing simulation practice.

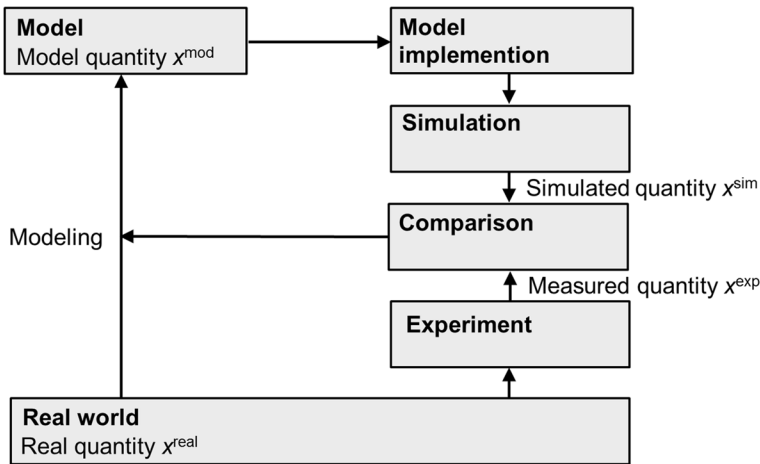
- Bad practices threaten reproducibility. Thus one part of the reproducibility problem is the normative question of what constitutes good simulation modelling practice. Good practice, however, does not guarantee reproducibility. This makes the problem philosophically interesting and brings in the second factor.
- Demanding complete reproducibility would be asking too much. It does not make sense setting up criteria that cannot be fulfilled anyway. What, therefore, is an adequate level of reproducibility?

Finding an adequate account of reproducibility will require investigating into the methodology of simulation modelling, and the concept of numerical solution. A common definition says that simulation is the numerical solution of a theoretical (or mathematical) model. For instance, the entry on ‘computer simulation’ in the Stanford Encyclopedia (Winsberg 2018) discusses a narrow, a broad, and an alternative definition, all of which specify simulation as numerically (or approximatively) solving mathematical equations of a model.

Our main point is that this definition is misleading. It is not strictly false, but it is tenable only under a sophisticated understanding of what “numerical solution” means—an understanding that includes a significant difference to the normal mathematical concept of solution. While complete reproducibility is a core property of mathematical solutions, numerical solutions are different. Not only numerical errors, but also social and pragmatic effects of implementation undermine reproducibility. We argue that simulation modelling can attain reproducibility only to a certain degree because it is working with numerical solutions (in a sense to be specified). Here is an outlook on how we examine simulation modeling.

Section 2 briefly presents the steps involved in simulation modeling. This account highlights that the translation from a theoretical (mathematical) model to a simulation model involves a whole series of modeling steps. These steps are mutually intertwined and together they constitute a modeling task of their own. Claims like this have been made repeatedly in the philosophy of simulation (see, e.g., Winsberg 1999 on layers of models, or Lenhard 2007 on the significance of discretization). It is important, however, to take a close look and expand the picture of simulation modeling. One point that becomes visible then is that modelers optimize numerical solutions at the cost of reproducibility.

Section 3 contains the main part of our analysis and argument. We discuss four aspects of the expanded picture: Sect. 3.1 the iterative nature of simulation and the



**Fig. 1** Schema of simulation modeling, including a feedback loop (from Hasse and Lenhard 2017). Courtesy of H. Hasse and J. Lenhard

resulting properties of numerical solutions, Sect. 3.2 parameters of the implementation, like adaptive grids, that loosen the connection between the theoretical model and the simulation, Sect. 3.3 limitations of modification, especially those that are due to proprietary software, and Sect. 3.4 the identity of what is getting solved. What gets numerically solved is a convolution of many modelling steps—not the theoretical (mathematical) model.

Section 4 summarizes the findings. Numerical solutions are a kind of compromise and limit the level of attainable reproducibility. An adequate degree of reproducibility in simulation should maintain a normative dimension, but should not ask for the impossible.

## 2 Simulation Modelling—Expanding the Picture

Computer simulations often build on a mathematical model. There are cases like agent-based systems, or cellular automata in which the role of a theoretical mathematical model is unclear. However, those cases in which there is such model are a class of great importance that we take as the standard case.<sup>3</sup>

A typical example is fluid dynamics where the same mathematical core is used when designing an aircraft wing, planning the pipes in an industrial plant, or when simulating the atmosphere. Consider a picture like the one displayed in Fig. 1.

There, we start with a quantity  $x^{\text{real}}$  in the real world, or less metaphysically, the target system that we want to model. This could be the vorticity of a flow. Although

<sup>3</sup> Readers interested in taxonomic intricacies might want to have a look at how simulation is defined in handbook or encyclopedia articles, see Lenhard (2016), Parker (2013), or Winsberg (2018).

the figure does not display the term ‘theory’, how one conceptualizes relevant phenomena, like vorticity or eddies, is already a theory-laden issue. Hence the entire figure should be read against the background of theory. The Navier–Stokes equations, a system of partial differential equations, are commonly held to present a very accurate model of fluid flow based on continuum mechanics and the conservation of mass, momentum and energy. The corresponding entity in the theoretical model is  $x^{\text{mod}}$ —think of the mathematically defined vorticity. Although the theoretical model is formulated in the language of mathematical calculus, it is not tractable with it. In other words: researchers cannot mathematically solve the equations and obtain  $x^{\text{mod}}$ ; it is even an open mathematical problem whether the 3D Navier–Stokes equations have a smooth mathematical solution over time if the initial conditions are also smooth. Instead, the model is implemented on a computer and simulations are carried out. Implementing the model requires a discrete version (there are many of these) that can only be some approximation of the theoretical model. A discrete model still needs to be implemented on a computer, including algorithms and describing them in a concrete software program. The resulting simulation model—or executed simulations—then yield a quantity  $x^{\text{sim}}$  (the simulated vorticity) that can be compared eventually to the results of experimental studies of the real world  $x^{\text{exp}}$  (measuring vorticity). Such comparison is neither trivial nor direct. Setting up an experiment and specifying procedures that measure the target quantity often is a demanding task. In general, neither  $x^{\text{real}}$  nor  $x^{\text{mod}}$  can be accessed; only the corresponding properties  $x^{\text{exp}}$  and  $x^{\text{sim}}$  can be compared. It might even be difficult to determine their difference.

This picture of simulation modeling is standard, but very compressed. Any such picture misses details, much like any map misses details of the landscape, but the standard picture misses parts that are crucial in our context. The critical point is that the path from the theoretical model (upper left) to the simulation model is described here as “model implementation”. It appears to consist of merely one step. But there is an entire series of steps. Without looking at them, talking of numerical solutions must miss the point. The picture of simulation modeling must therefore be expanded.

Before we turn our attention to the expanded picture, we would like to observe that Fig. 1 entails a feedback loop that is of critical importance when building simulation models. All kinds of modeling can include a feedback loop, but making intense use of the loop is practical only when the loop works swiftly enough, i.e. modifying one element and then repeating all steps (calculations) is fast enough. This loop makes adjustable parameters a versatile tool. If, for instance, some simulation deviates from known experimental results, one can insert parameters whose adjustment can help to narrow down the deviation. Using adjustable parameters to optimize model performance requires marching fast and often through the feedback loop. Hence the loop invites modelers to combine top-down (starting from the theoretical model) with bottom-up (exploring parameter assignments) approaches (cf. Hasse and Lenhard 2017; Lenhard 2016). Figure 2 attempts to name the steps that lead from  $x^{\text{mod}}$  to  $x^{\text{sim}}$ .

In the following, we concentrate on some of the aspects that become visible in the expanded picture and that are pivotal for our topic of reproducibility and numerical solution.

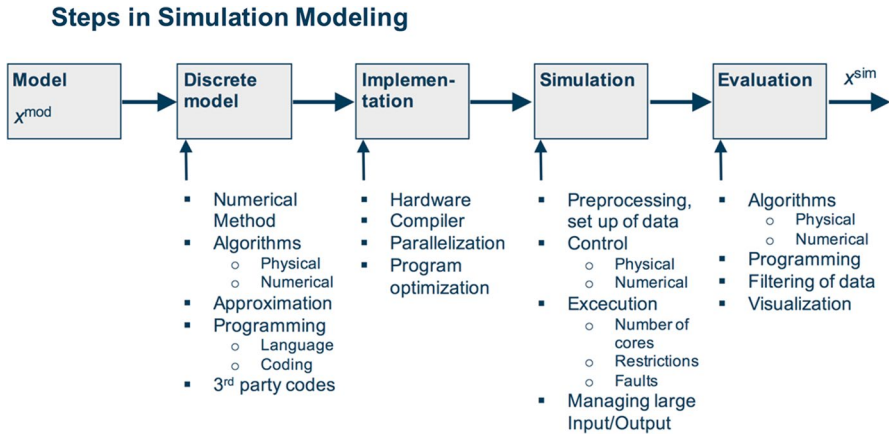


Fig. 2 An expanded picture detailing the steps between  $x^{\text{mod}}$  and  $x^{\text{sim}}$

### 3 Four Elements of the Numerical Part of Modeling

The expanded picture of Fig. 2 displays a number of steps that lead from  $x^{\text{mod}}$  to  $x^{\text{sim}}$ . The following discussion clarifies what is wrong with describing this pathway as “solution”.

#### 3.1 Iterative Nature

The digital computer has a special strength; it works through an iterative algorithm with high speed. Before the electronic computer, iterative algorithms were known, but their scope of application was very restricted. The calculus illustrates the point. Important properties of a mathematical function are defined along a path of iterations, just as a series of linear approximations on the nodes of smaller and smaller triangles describes the gradient of a function in 2D at one point. The calculus has replaced the iterations with one single operation, namely building the derivative, and thus has made the operation tractable. There are other cases, like the Raphson–Newton or the Euler methods that iterate numerical schemes. Their utility is that they yield an approximative result after not too many iterations (for human beings with great but finite skills and concentration span).

The computer changed the game. Iterating an algorithm is a straightforward and easy task to program (though the details may lead to substantial theoretical matters). Iterative algorithms render approximation strategies feasible that before had been intractable. This opportunity invites—and thereby guides and attracts—new developments, i.e. new approaches that make fuller use of iteration. Whether iterative procedures get to the “correct” solution as a limit, or whether they plausibly approximate it, remains a key question. By and large, linear systems can be solved (we come back to the notion of solution in this section) via direct (non-iterative) methods, whereas nonlinear systems demand iterative methods.

We would like to point out that two types of iteration are in play here. Iterative algorithms repeat a procedure, taking the result of the previous loop as an input. We call this *a-iteration*. Additionally, the feedback loop in the modeling process introduces an iterative element in the building of the model, including the adjustment of parameters. Here, the loop is iterated, but this can be different from iterating an algorithm, because depending on the result, modelers will learn and adapt. What they do next depends on the previous loop in a potentially complicated way.<sup>4</sup> The feedback loop helps to *control* the modeling process. We call this type of repeating the loop *c-iteration*. Examples for *c-iterations* are exploring parameterization schemes, adapting physical parameters, the number of particles included in a model, etc. We admit this terminology lacks elegance. Our excuse is that we do not want to propose *a-* and *c-iteration* as new terms. We just want to make clear that there are different types of iteration in play.

How does the highly iterative nature of numerical procedures affect the concept of solution? If you have an equation  $e$  with a variable  $x$ , and you replace  $x$  by a certain term and find that the equation holds, then this term solves  $e$ . This concept seems to be well-defined (the simple case we just described is sufficient) and hardly changeable at all.

A basic but important point is that only machines can do large numbers of iterations. And solutions generated by approximation procedures differ significantly from what is called a closed-form solution. For example, the closed-form solution might have algebraic or physical properties that the approximate solution does not have. A historical example shows the advantages of solutions. Galilei determined the ballistic curve of a cannonball. According to his model, it describes a parabola, i.e., a quadratic equation—and it is not too difficult to solve this equation. The solution, also called closed-form solution, will depend on initial conditions, like the angle of the gun barrel. That solution gives a handle on variations, too. How does varying the initial angle change the curve? Etc. Moreover, inverting the solution is possible: how should the artillerymen act if they want to hit a target? This is a primary point for why mathematical modeling might be useful. It opens up the opportunity for predictions and for further analysis of potential variations. Investing work into the solution is economical because the solution covers a whole set of situations.

By the way, we chose the military example because it played an important role historically: Computing ballistic curves has been a main motivation for developing numerical techniques. Furthermore, it nicely illustrates the fact that the predictive qualities of mathematical models and their solutions do not imply that such models would be accurate. It turned out that real cannonballs do not at all follow a parabola, since effects of friction, temperature, etc. change the mathematical form of the curve in complicated ways.

Independently of the model's being accurate or inaccurate, the situation looks quite different when we examine numerical solutions rather than closed form solutions. Numerical solutions in general satisfy the equation they solve only approximately, with an error, which might not be simple to estimate. But this description

---

<sup>4</sup> If adjusting parameters is automated, it can be conceived as an *a-iteration*.



requires clarification. If the initial model is a set of differential equations (think of the Navier–Stokes equations), the discretized version (numerical method) is a set of difference equations. Nonlinear difference equations are normally not solvable in closed form, but they are straightforwardly handled by iteration. Such difference equations solve the initial model approximately, but doing so requires a series of additional steps (see Fig. 2). Thus it is an additional problem how well a simulation accomplishes the approximation in practice. One typical question is how quickly the approximation improves with the number of iterations. We want to avoid getting technical here.

Although in quantitative terms, the numerical solution might approach the correct solution (of the mathematical model), in practice the properties remain very different. Inverting the result is not possible with typical numerical solutions because there is no backward pathway through the calculations. How strongly does the solution depend on the initial conditions? Can it vary largely even if the conditions vary only slightly in the case of an ill-posed problem? Questions like these involve not only the numerical methods and algorithms chosen, but also their concrete implementation. Answering these questions therefore requires iteration on top, i.e. changing the parameters and running the simulation again.

Iteratively sounding out the space of solutions is a very different strategy from obtaining a closed form mathematical solution, if there is one. Numerical methods almost never get it precisely right. More importantly, they do not come with the additional helpful properties (variation, inversion, conservation of certain properties). Thus, a numerical solution provides knowledge about the mathematical model (in the sense of section two) that is quite different from what a traditional mathematical solution provides. One can see this difference when considering possible failures. Considering failures is the flip side of considering solutions.

We discern two types of failures (ignoring the fact that the space of failures is much larger). For type one, there is a remedy if the adequate tool is available. For type two, there is no remedy. An example for type one failures is the numerical weather forecast that L.F. Richardson obtained for part of Germany. He worked during the First World War when electronic computers did not exist. Back then, the word computer referred to human beings. Richardson set up a numerical scheme (system of difference equations) that would describe how meteorological conditions process in time (based on physical laws). The model had been formulated as a system of (continuous) intractable partial differential equations by the meteorologist Wilhelm Bjerknes in 1903. It took Richardson 6 weeks to compute a 6 h forecast for one single cell in Germany, for known weather conditions of the year 1910. Alas, it turned out that the numerical forecast was utterly wrong. Later researchers analyzed the reasons, but it is not fully clear whether imprecise initial conditions or a misleading discretization (Richardson did not know the Courant-Friedrichs-Lewy conditions) caused the failure. Richardson writes that he was motivated by a fantasy: an intractable mathematical model could be made tractable via organizing computational power.

“After so much hard reasoning, may one play with a fantasy? Imagine a large hall like a theatre, except that the circles and galleries go right round through the space usually occupied by the stage. The walls of this chamber are painted to form

a map of the globe. The ceiling represents the north polar regions, England is in the gallery... A myriad computers are at work upon the weather of the part of the map where each sits, but each computer attends only to one equation or part of an equation." [Computers exchange values with their neighbors. On a tall pillar in the middle] "sits a man in charge of the whole theatre; he is surrounded by several assistants and messengers. One of his duties is to maintain a uniform speed of progress in all parts of the globe. In this respect he is like the conductor of an orchestra in which the instruments are slide-rules and calculating machines." (Richardson 1922, pp. 219–220)

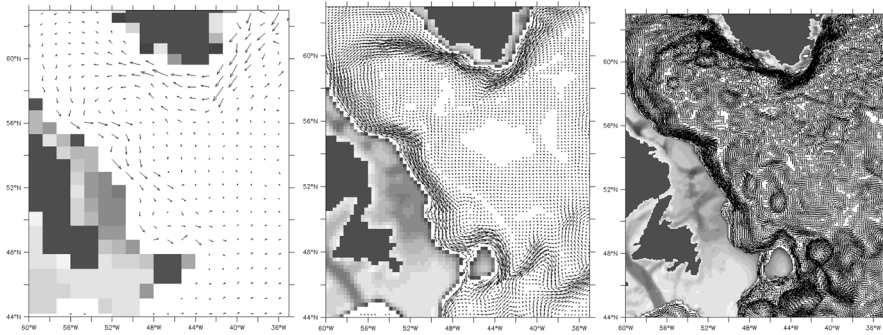
No matter what exactly caused Richardson's failure, with the right tool at place (digital electronic computer with sufficient power, plus experience with rules for discretization), his task would have become a doable one. He anticipated even parallel computers.

Type two failures are different: There simply is no remedy in practice. Such failures are virtually ubiquitous in simulation. One example out of many is molecular dynamics simulation where researchers model the behavior of molecules by a large number of particles that interact according to certain rules (force fields). How many particles should they model? Is there a reason why 1 million particles are sufficiently many in molecular simulations compared to the Avogadro-constant of  $6.022 \cdot 10^{23}$ /mol? Obviously, this number has to be assigned before the iterative algorithm can start. It is an assumption that cannot be justified based on the mathematical model. Or maybe it can, but the justification is in typical cases unknown. It is a pragmatic assumption: Take a big number, but not so big that it is slowing down the iterative algorithm too much. That means the number is strongly dependent on the available computational capacity. Typically, simulation models will entail failures of these types. They are simply unavoidable. Not every part of a simulation model (in the expanded picture from Sect. 2) can be theoretically motivated. After healing Richardson's failure, there surely remain several failures of type two in meteorological models. Investing more computation in general helps and makes the approximation better. But how good precisely? In typical cases, the exact relationship between a numerical solution (the simulation) and the exact solution (of the theoretical model) is unknown. A numerical solution can never get it right.

Numerical solutions are different from traditional mathematical ones because the former bring in pragmatic aspects and relax the connection to "the" correct solution. The following sections will add analyses that basically support these findings.

### 3.2 Parameters

We stay with the example of fluid dynamics. The Navier–Stokes equations are an (in)famous set of partial differential equations (PDEs). They are famous because they represent the dynamics in a very comprehensive way. At the same time, they are infamous because mathematically solving the PDEs is of utmost difficulty. Already the question whether a continuous solution exists (not to give this solution) has made it into the "Millenium" list, i.e. the shortlist of important unsolved mathematical problems. The problem of practically obtaining this solution (if it exists)



**Fig. 3** a–c Oceanic circulation in the Labrador Sea. Increasing resolution from left to right: 4/3, 1/3, 1/12 degree. From Böning et al. (2002, pp. 11–13)

is on a list that itself does not yet exist. However, simulation methods have made fluid dynamical problems tractable. The Navier–Stokes equations can be solved numerically (with all caveats the Sect. 3.1 discussed regarding the notion of solution). Since simulation methods are more or less a requirement when working on practical fluid dynamical problems, and since these problems are highly relevant in many different fields, fluid dynamics is a prime example of simulation. Moreover, it provides a motivation for the standard definition of simulation (see introduction) as numerical solution.

A most important phenomenon of turbulent flow is the emergence of eddies. Their behavior strongly influences the global properties of the flow. That eddies emerge is inherent in the continuous Navier–Stokes equations. Simulations require the discretization of space and time. Obviously, eddies strongly depend on the resolution of the discrete grid. Depending on the order of the scheme no grid can resolve phenomena that are essentially smaller than grid size, just like no net can catch fish smaller than the grid between knots. Figure 3 shows an example of a fluid dynamical simulation, the oceanic circulation in the Labrador Sea. The highest resolution has been computed at the DKRZ, Hamburg, Germany, and has had a runtime of 100–150 h CPU per processor and model year (run in year 2002, 19 Gflop on 8 cores, today's machines might be much faster, but do not change the principle).

Obviously, the eddies shape the picture of the circulation with increasing resolution. Consequently, the calculated fluid dynamical properties depend on the way the grid is refined. The crucial question then is, how the grid refinement should look to produce a realistic picture (the Kolmogorov length scale for atmospheric motion might be measured in centimeters). It is very likely that also future computers are not able to resolve all eddies of the flow. Is this a problem of the second type (Sect. 3.1)?

There are at least two different, but connected problems. One is obtaining a good approximation on a given grid, the other is deciding on which grid is adequate. The first problem is a generic one when moving from continuous mathematical models (like the Navier–Stokes equations) to discrete ones. The behavior of the discrete model will be different from the continuous one. The former will introduce artefacts relative to the initial continuous model, irrespective of whether the initial

model is adequate or not. One example is that the discrete version lacks stability when it comes to high frequencies. This problem has been acknowledged from the very beginning on, when researchers started to explore the possibility of numerical simulation. John von Neumann and others, while working in the Manhattan project, proposed to introduce an “artificial viscosity” that compensates the artificial effects caused by the grid (Winsberg 2003 discusses this case). Refining the grid will also change this artificial viscosity. The idea is that if the grid gets finer and finer, the artificial viscosity tends to zero. Ideally, an infinitely fine grid (not feasible in computational practice) will lead to zero artificial viscosity.

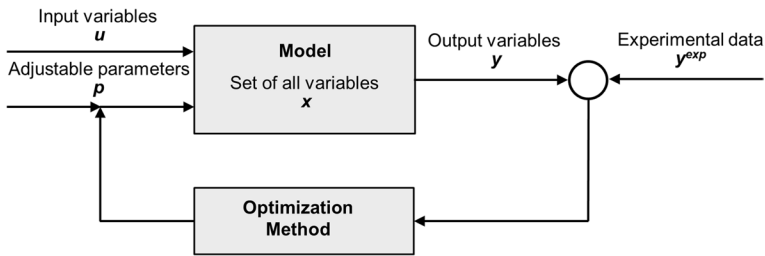
As a rule, deviation between theoretical and simulated behavior, i.e. between  $x^{\text{mod}}$  and  $x^{\text{sim}}$ , is considered unwanted. A very common remedy against such unwanted effects is amending simulation models with artificial elements that compensate for the deviations.<sup>5</sup> In the picture of simulation modeling discussed in Sect. 2, artificial viscosity appears as a parameter (or set of parameters) that can be adapted. Depending on the parameter assignment, the numerical solution becomes better or worse compared to known cases or measurements. Such compensation can have drawbacks. In our example, artificial viscosity might collide with the inherent viscosity of the Navier–Stokes equations and influence phenomena like eddies in unforeseen ways.

The exact dependence of the fluid dynamical properties from the chosen grid refinement is usually not known, but has to be explored (via *c*-iteration). In practice, refinement strategies are often guided by expert knowledge that resembles the knowledge of an artisan. At the same time, and because of the informal character of the grid adaptations, one hardly finds explicit considerations in published papers. They usually report only which grid (refinement) has been used, seldom do they mention which other refinements had been explored, and practically never can one read about reasons why some refinement was superior to others. This is no wonder, because formal justification is normally not available.

The second problem asks which grid is adequate. Figure 3 depicts that the behavior is strongly dependent, if not dominated by eddies. Would further refinements lead to a significantly different picture, and if so, would these refinements be helpful given the simulation purpose? Questions like these are hard to decide. When interesting properties do not change with further refinement, i.e. are robust, this outcome would strongly indicate that the grid is sufficiently fine. Ideally, modelers could validate for independence from grid parameters. That would mean they check whether simulation results are independent of the exact parameterization. But such tests are rarely performed, because the programming community considers parameter-independence a matter of experience and fingertip knowledge on the part of model builders. In many circumstances, the grid resolution is already as fine as possible, given restrictions in time and money. Hence testing further refinements is practically not possible.

---

<sup>5</sup> Lenhard (2007) examines another case where a pioneer of simulation met resistance because of the artificial elements he advocated.



**Fig. 4** Parameter adjustment as a feedback loop (from Hasse and Lenhard 2017). Courtesy of H. Hasse and J. Lenhard

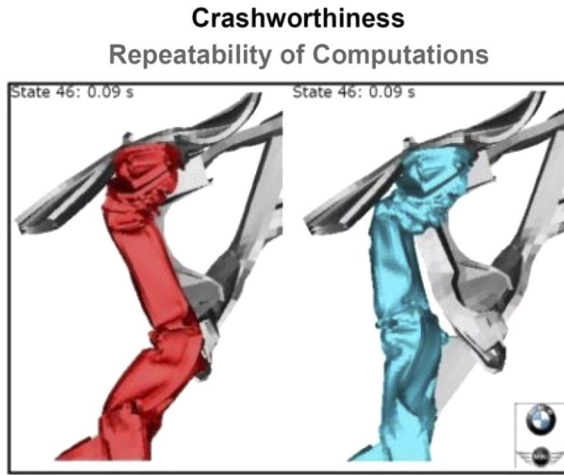
The example of artificial viscosity presents a generic dilemma of simulation. The artificial viscosity should be as small as possible, because it is an artificial remedy and as such indicates that the model needs such remedy. On the other side, if a bigger artificial viscosity (or artificial whatever) leads to better results, this outcome normally trumps the first aspect. A numerical solution, like the one displayed in Fig. 3c, will balance both counteracting effects. Hence it will rest on a *compromise* and therefore has an essentially pragmatic nature. This lesson applies also to other existing numerical approaches (“upwinding”) that need no artificial viscosity.

Although considerations like these are not part of physics or fluid dynamics, they are inevitably part of simulation modelling. Most questions about parameter assignment are not related to the underlying mathematical model, and thus to the extent that parameters influence numerical solutions, these solutions do not “solve” the mathematical model. There are many reasons why parameters enter the simulation model along the way displayed in Fig. 2. One very principled perspective is the following: All models idealize and omit some of the relevant details. Such idealization might be even desired. Additionally, knowledge is always incomplete and hence one cannot model the full target system. Thus, when parameters are adjusted, they fit  $x^{\text{sim}}$  best fits to  $x^{\text{exp}}$  (cf. Fig. 1) and thus compensate for missing details and missing knowledge. Again, if a simulation is a solution, finding this solution is like finding a compromise.

### 3.3 Limitations of Modification

There is a serious technical limitation to modification. Parameters like those governing grid refinement span a parameter space with many different parameters that is much too large to be explored exhaustively by numerical simulation. Finding the best or most adequate assignment of values is typically an unsolvable problem itself. Finding some assignment that is good enough is the best what one can hope for. Hasse and Lenhard (2017) discuss adjustable parameters in more detail. Figure 4 shows the basic task when looking for a good adjustment.

Often, modelers do not search for an adjustment by hand, but employ formal optimization methods. They might outsource an optimization task to a numerical solver that will handle parameters according to a formal procedure. The resulting parameter assignment works best for formal reasons, but motivations for why this is so are



Identical FEM models; identical software / hardware  
and nevertheless largely differing computational results

**Fig. 5** Reprinted from Duddeck (2007). Courtesy of Fabian Duddeck

often unavailable. In other words, such procedures create epistemic opacity because parameters tend to lose their interpretation.

Here, we concentrate on another reason that limits modification. The car manufacture industry uses simulated crash tests when designing and testing their new cars. When repeating empirical crash tests, car bodies will never deform in the very same way. But users (and modelers) expected this would happen in *simulated* crash test. For them, it came as a surprise that repeating the same simulation experiment (crash test) can lead to significantly different outcomes. Figure 5 shows an example. The observed differences motivated a research project that analyzed these unwanted variations.

A team conducted simulations with software packages that are standard in finite element simulations of crash tests, like Abaqus or LS-Dyna. They found strong indications that several factors worked together in producing variations that had no counterpart in the target system, i.e. were artefacts of the simulation. By modifying the input, they could analyze the sensitivity of the various packages. However, this story found a premature end because these packages are 3<sup>rd</sup> party code that is proprietary, and the modelers were not allowed to analyze the source code. Hence they could not do anything about the observed or assumed flaws. If simulation modelers find themselves under pressure to improve the software without access to the source code, they can resort to adding some extra artificial measures that are called “kluges” (also spelled “kludges”) in software slang. Kluges increase opacity, however, because there is no reason why they are as they are except for the tested overall performance. Even having access to the source code, however, would not lead to a quick solution. One would first have to become acquainted with important parts of the code. People who have ever tried to understand the logic behind code written by

someone else, are immediately aware how big the problem is. One indication is that manuals of common software packages have often a 4-digit number of pages.

More important for these kinds of numerical tests is that the model for crashes is itself not well-posed. That means small changes in initial conditions may produce large variations in results. Even very accurate numerical simulations cannot achieve anything better than predicting these variations. The question then remains open, of whether these predictions are meaningful.

A third and different type of limitation comes from the connection between  $x^{\text{real}}$ ,  $x^{\text{mod}}$ , and  $x^{\text{sim}}$ . An example will make the point. In fluid dynamics (and other areas), there is a phenomenon called dispersion, i.e. velocities split and waves of different wave lengths propagate with different velocities. This is a “real” phenomenon (think of monster waves where waves of different velocities pile up at a certain moment) and hence a good simulation model should be able to produce this kind of behavior. At the same time, dispersion also emerges as an artefact of discretization. In this role, dispersion is not wanted. But how to discriminate the two parts or sources of dispersion? One option would be to maximize resolution everywhere, but this is not possible because of the criterion of economy (limited run time). Again, the art of finding a numerical solution partly consists in finding a compromise between factors that are not fully formalized.

### 3.4 A numerical Solution Solves What?

The crucial point has already been made. It is fruitful to see a simulation as a numerical solution, but one should keep in mind that this solution does *not* solve the (original) theoretical/mathematical model. Rather, the result is about an entire convolution of the theoretical model with all modelling steps and loops involved in Fig. 2 (for the moment we assume this figure gives a complete picture). In our previous example of fluid dynamics, the simulation does not numerically solve the Navier–Stokes equations as partial differential equations, but their specific, discrete version, plus the grid, plus the parameterization schemes, plus specifications of parameters.

Thus we encounter a twofold problem about how simulation is linked to numerical solution. First, the simulation does not solve the initial model. It rather solves a model that is a complicated set of assumptions and that is given only implicitly by the convolution of all modeling steps. It is not “the” model that runs on the computer. It is this complicated and implicit convolution that produces the results ( $x^{\text{sim}}$ ). It cannot be the unique correct solution (if it exists in a mathematical sense)—and, more importantly, speaking about approximation is often a thinly disguised hope rather than a clear modeling condition.

The second part of the problem is not about what is solved, but about how it is solved. Numerical solution starkly differs from normal mathematical solution. Our investigation revealed that numerical solutions, in contrast to mathematical ones, have a pragmatic character. Notions like tractability and the economy of time and money co-determine what counts as a numerical solution.

## 4 Conclusion

This paper started with the claim that the problem of reproducibility exists in simulation modeling and that examining this problem leads to re-thinking the concept of numerical solution and how simulation is related to it. Now we look backwards, starting from simulation and numerical solution. The usual picture assumes that a small number of steps lead from a theoretical mathematical model to a discrete model and further on to the final simulation. Practitioners as well as philosophers of science commonly refer to simulations as solutions, but this way of speaking is only accurate *if* the numerical method is stable under the conditions in questions *and* the simulation approximately solves the numerical method with sufficient accuracy. Investigating whether these conditions are met requires a detailed picture of the modeling steps involved. Only a picture of simulation modeling that, compared to the common picture, resolves many more steps can address the relevant numerical, technical, and social aspects.

Problems with interfaces between the modeling steps and the role of iteration (both a- and c-iteration) become visible only in the expanded mode. Simulation modeling interconnects quite different entities, the theoretical model and the computer simulation. The host of extra assumptions, artificial additions, parameterizations, and the effort to write a simulation program testify how demanding it is to construct this link.

Thus a main lesson is how important the expanded picture of simulation modeling is. That the picture used here (Fig. 2) might be incomplete only strengthens the case. Since the modeling steps are interdependent and of very different character, the outcome attains features of a compromise. This gives reason to disambiguate the concepts of “computer simulation” and “numerical solution”.

How does this conclusion relate to the problem of reproducibility? To expect bit-wise reproducibility would be naïve. Numerical analysts are aware of this fact which is even reflected in the official IEEE norms. The commonly supposed close link between simulation and solution motivates a high expectation concerning the level of reproducibility. Strictly proven mathematical solutions possess unquestionable and complete reproducibility. They might even count as the paradigms of reproducibility. However, since computer simulations can count as numerical solutions only if solution is understood in a pragmatic sense, a very high level of reproducibility is much less plausible. Not only numerical errors come in as factors of computer-related mathematics, but issues of the implementation bring in factors of social and pragmatic nature. We think it is important to reflect on this and other transformations that happen when simulation modelling replaces older approaches to mathematical modelling. This is a time when new scientific practices of simulation grow and practitioners have not yet agreed upon the standards they follow. Finding such standard requires having a clear conception of what can be reproduced (in the best case) and what never will be reproducible (even in the best case). Potentially, most problems of reproducibility can be alleviated by changing simulation practice.

Growing awareness that simulations might require new conceptions for confirmation and for measuring the quality of numerical solutions has led to interesting



developments. Numerical error analysis, for instance, shows that much could be achieved when modelers adopted a mathematical perspective on a larger number of modeling steps, i.e. on a larger fraction of the entire modeling process. How this works out in practical cases like car crash simulation remains yet to be seen. Another interesting concept is sensitivity analysis. It aims at giving some of the relevant information that researchers would get from a closed-form mathematical solution. Another example is working with so-called model ensembles, i.e. an entire class of models that vary in a controlled way. They are used mainly in climate science to test the robustness of model properties. Finally, there is a normative dimension: what are adequate goals that should guide the modelers? This much should be clear: Bitwise reproducibility does not make sense as a goal because it is not attainable. If simulation models are unable to reproduce a result and if (a big if) the sources of errors have been eliminated, the remaining lack of reproducibility can indicate a serious point. Maybe the problem is ill-posed, i.e. great sensitivity from initial conditions is due to the (mathematical) problem itself, not due to erroneous implementation. In these cases, the problem does not allow a meaningful numerical solution and, therefore, the quality of simulation-based predictions is questionable. An important question for mathematics therefore is: what is maintained even in the ill-posed case? Often, obtaining *the* solution is not even the goal. Giving some handle, guiding a decision etc. can work with much less. Whether this is a good or bad thing, is a political question.

**Acknowledgements** We like to thank three anonymous reviewers for useful suggestions and Nicholas Danne for his support in approximating our text to English language.

## References

- Böning, C., Beismann, J.-O., Biastoch, A., Czeschel, L., & Degg, J. (2002). Ozeanische Aufnahme anthropogener Spurengase: Realistische Darstellung des Effektes mesoskaliger Prozesse in Zirkulationsmodellen. *Presentation at DKRZ WLA-Workshop*, 24(10), 2002.
- Collins, H. (1985). *Replication and induction in scientific practice*. Chicago: Chicago University Press.
- Duddeck, F. (2007) Survey on robust design and optimization for crashworthiness. In *EUROMECH colloquium 482: Efficient methods for robust design and optimization*. Queen Mary, University of London, London, UK.
- Fillion, N. (2017). The vindication of computer simulations. In J. Lenhard, M. Carrier (eds.) *Mathematics as a tool* (pp. 137–55). Boston Studies in History and Philosophy of Science 327. New York: Springer.
- Fillion, N., & Corless, R. M. (2014). On the epistemological analysis of modeling and computational error in the mathematical sciences. *Synthese*, 191, 1451–1467.
- Hasse, H., & Lenhard, J. (2017). Boon and bane. On the role of adjustable parameters in simulation models. In J. Lenhard, & M. Carrier (eds.) *Mathematics as a tool. Tracing new roles of mathematics in the sciences*. Boston Studies in the Philosophy and History of Science (Vol. 327, pp. 93–115).
- Ioannidis, J. P. (2005). Contradicted and initially stronger effects in highly cited clinical research. *JAMA*, 294(2), 218–228.
- Kaminski, A., Resch, M., and Küster, U.: Mathematische Opazität. Über Rechtfertigung und Reproduzierbarkeit in der Computersimulation. In A. Friedrich, P. Gehring, C. Hubig, A. Kaminski, & A. Normann (eds.) *Jahrbuch Technikphilosophie* (Vol. 4, pp. 253–277). Nomos Verlag, 2018.
- Lejaeghere, K., Bihlmayer, G., Björkman, T., Blaha, P., Blügel, S., Blum, V., et al. (2016). Reproducibility in density functional theory calculations of solids. *Science*, 351, 3000.

- Lenhard, J. (2007). Computer simulation: The cooperation between experimenting and modeling. *Philosophy of Science*, 74, 176–194.
- Lenhard, J. (2016). Computer simulation. In P. Humphreys (Ed.), *Oxford handbook in the philosophy of science* (pp. 717–737). New York: Oxford University Press.
- Ludwig, T. (2017). Reproducibility in science, computer science and climate science. News from Computational Climate Science. Presentation in: Leogang, 08.03.2017.
- Open Science Collaboration. (2015). Estimating the reproducibility of psychological science. *Science*, 349(6251), 4716.
- Parker, W. S. (2013). Computer Simulation. In S. Psillos & M. Curd (Eds.), *The routledge companion to philosophy of science* (2d ed., pp. 135–145). New York: Routledge.
- Richardson, L. F. (1922). *Weather prediction by numerical process*, 1st edn. 1922, Cambridge UP; cited according to second unaltered and unabridged edition with new introduction by Sydney Chapman, 1965, Dover.
- Schappals, M., Mecklenfeld, A., Kröger, L., Botan, V., Köster, A., Stephan, S., et al. (2017). Round Robin study: Molecular simulation of thermodynamic properties from models with internal degrees of freedom. *Journal of Chemical Theory and Computation*, 13, 4270–4280. <https://doi.org/10.1021/acs.jctc.7b00489>.
- Solomon, M. (2011). Just a paradigm: Evidence-based medicine in epistemological context. *European Journal for Philosophy of Science*, 1, 451–466.
- Winsberg, E. (1999). Sanctioning models: The epistemology of simulation. *Science in Context*, 12(2), 275–292.
- Winsberg, E. (2003). Simulated experiments: Methodology for a virtual world. *Philosophy of Science*, 70, 105–125.
- Winsberg, E. (2018). Computer simulations in science. In Edward N. Zalta (ed.) *The stanford encyclopedia of philosophy* (Summer 2018 Edition), forthcoming. <https://plato.stanford.edu/archives/sum2018/entries/simulations-science/>.

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.