Chapter 43
The Multidimensional Epistemology of Computer Simulations: Novel Issues and the Need to Avoid the Drunkard's Search Fallacy

Cyrille Imbert

Abstract Computers have transformed science and help to extend the boundaries of human knowledge. However, does the validation and diffusion of results of computational inquiries and computer simulations call for a novel epistemological analysis? I discuss how the notion of novelty should be cashed out to investigate this issue meaningfully and argue that a consequentialist framework similar to the one used by Goldman to develop social epistemology can be helpful at this point. I highlight computational, mathematical, representational, and social stages on which the validity of simulation-based belief-generating processes hinges, and emphasize that their epistemic impact depends on the scientific practices that scientists adopt at these different stages. I further argue that epistemologists cannot ignore these partially novel issues and conclude that the epistemology of computational inquiries needs to go beyond that of models and scientific representations and has cognitive, social, and in the present case computational, dimensions.

Keywords Computer simulations · Novelty · Epistemology · Validation · Goldman · Consequentialism · Social epistemology · Random numbers · Computational values · Monte Carlo · Random number generator · Modeling norms · Modularity · Opacity · Verification · Reproducibility · Models · Invisibleness of failure · Naturalism

The drunkard's search or streetlight fallacy corresponds to a type of situation where people search at the easiest site, even if what they are searching for is unlikely to be there. Typically, the drunkard searches for her keys under a streetlight even if they were lost somewhere else.

C. Imbert (✉)
Archives Poincaré, CNRS, Université de Lorraine, 91 avenue de la Libération - BP 454., F-54001 NANCY Cedex, France
e-mail: Cyrille.Imbert@univ-lorraine.fr

C. Beisbart and N. J. Saam (eds.), *Computer Simulation Validation*, Simulation Foundations, Methods and Applications,
https://doi.org/10.1007/978-3-319-70766-2_43

43.1 Introduction: Computer Simulations, a Revolutionary Epistemology?

The search for innovation and novelty are major goals across social fields. Unsurprisingly, when new technologies, artifacts, techniques, methods, practices, perspectives, or issues are developed, bold statements about their potential impacts are made. So-called "revolutions" or "turns" are regularly announced across science and it is legitimate to be methodologically cautious about such claims.

It is hardly controversial that computers have largely transformed science and help to extend the boundaries of human knowledge. Yet, it might be the case that computers merely bring about more inferential power but that concerning how scientific results are justified and come to be trusted, science is left unchanged by the computational revolution.

Philosophers of science have had a long-standing tradition of analyzing experiments, theories, and scientific reasoning. However, specific epistemological analyses of simulations did not develop until the 1990s with work by philosophers like Paul Humphreys, Eric Winsberg, or Manfred Stöckler, historians of science like Peter Galison, scientists interested in philosophical issues like Fritz Rohrlich, or scholars at the crossroads of several fields like Evelyn Fox Keller. These different authors mostly agreed that computational methods not only provided a new powerful way to practice science, but also did not match existing categories and called for specific and novel analyses, above and beyond those concerning experiments, theories, or models.

In a thought-provoking and conservative article, Roman Frigg and Julian Reiss stood against this move and argued that claims about the novelty of computational science were overblown and ill-grounded and that there was no more to the epistemology of simulations than the epistemology of modeling (Frigg and Reiss 2009). Making final and flawless contributions is difficult for those who pioneer in a field and various aspects of Frigg and Reiss's jubilant refutation were convincing. The need to guard against the lure of apparent novelties was later confirmed, for example, by the criticism by Barberousse and Imbert (2013) of revolutionary claims about cellular automata based simulations (a case that was recurrently used in favor of novelty claims) or by the sober and deflationary analysis by Beisbart of the deeply argumentative nature of simulations despite their genuine similarities with experiments (Beisbart 2018, see also Barberousse et al. 2009). However, Frigg and Reiss were not content to refute claims about the novelty of specific aspects of simulations. They extrapolated that simulations "raise few if any new philosophical problems" (593) and suggested considering the literature on simulations "as contributing to existing debates about, among others, scientific modeling, idealization or external validity, rather than as exploring completely new and uncharted territory" (595).

Paul Humphreys quickly responded that this general non-novelty claim was simply false. In (Humphreys 2009), he highlighted that issues such as the epistemic opacity of computational processes, the importance of syntax, complexity questions, or the specific role of time in simulations all make the epistemological and semantic analysis of computational science novel, beyond genuine overlaps with existing philosophical analyses of science.

The present chapter focuses specifically on the epistemology of simulations, how their results are validated, and whether the problems that arise in this context are novel. Frigg and Reiss' debunking paper was a sanitizing contribution. Nevertheless, I will also argue that their main conclusion is false because simulations raise new epistemological questions or raise traditional questions that require novel or specific answers for simulations.

I devote Sect. 43.2 to philosophical preliminaries: I first discuss how the notion of novelty should be cashed out here and argue that using a conceptual framework similar to the one used by Goldman to develop social epistemology is appropriate for the investigation of the present question. In Sect. 43.3, I list computational, mathematical, representational, and social loci on which the validity of simulation-based belief-generating processes hinges. Additionally, I emphasize that their epistemic impact depends on the practices that scientists adopt to face these problems. I further argue in Sect. 43.4 that epistemologists cannot ignore these issues and conclude in Sect. 43.5 that this analysis agrees with those which emphasize that the epistemology of science needs to go beyond that of scientific representations and has cognitive, social, and here computational, dimensions.

43.2 Methodological and Conceptual Preliminaries

First, the notion of novelty should be clarified if it is to frame the discussion. What is scientifically novel is contingent upon which claims, theories, or perspectives have been defended within a field. Thus, the real issue is whether an object of inquiry should be analyzed along the same lines as other objects. The difference can be illustrated as follows. In the context of computer simulations, it is blatantly obvious that complexity and computational resources must be taken into account to analyze the constraints that frame computational inquiries. Accordingly, focusing on what is possible in practice (Simon 1957; Humphreys 2004; Wimsatt 2007) and emphasizing the importance of the scarcity of resources for agents is appropriate. However, resource-boundedness is a general constraint that frames *both* computational and noncomputational inquiries. Humphreys suggests the general epistemological principles that "it is the invention and deployment of tractable mathematics that drives much progress in the physical sciences" and that "most scientific models are specifically tailored to fit, and hence are constrained by the available mathematics" (see Humphreys 2004, 55–56 and Barberousse and Imbert 2014 for a detailed discussion). Still, it so happens that in existing discussions about models, complexity issues merely arise as a peripheral point to justify the need to make approximations. In brief, whereas the development of computational science somewhat relaxes computational constraints and resource-boundedness constitutes a much more restrictive straightjacket for traditional noncomputational inquiries (see again Barberousse and

Imbert 2014), quite paradoxically, the need to adopt a bounded-resource perspective is blatant and apparently novel in discussions about computational inquiries.

Also, the notion of the novelty of questions, discussions, or of "uncharted territory", partly places novelty in the wrong location. While some aspects of computer simulations trigger new questions (e.g., concerning code or the epistemic opacity of computational processes), others raise questions of types that are already analyzed by epistemologists but need no less epistemologically revolutionary answers. For example, the role of human faculties in the architecture of human knowledge is a central issue in mainstream epistemology. This role sometimes changes. Over the centuries, the development of measurement instruments transformed empirical science and made it less dependent on our senses. However, till the advent of computers, methods, languages but also objects of inquiries were adapted to human reasoning and inferential abilities, the reliability of which was crucial to that of scientific results. The development of computer-assisted science keeps transforming this situation (Humphreys 2009, 616). Computers carry out increasing parts of inferential processes and scientific inquiries are less adapted to our inferential capacities in their objects and methods. However, humans remain the architects of these inquiries, the devisers and warrantors of methods and instruments, and the recipients of scientific results. In brief, science is no longer *human-tailored* but remains *human-centered* (Imbert 2017, 771), and we are faced with the "*anthropocentric predicament*, of how we, as humans, can understand and evaluate computationally based scientific methods that transcend our abilities" (Humphreys 2009, 617). Overall, the development of computational science requires reexamining the place of human faculties in knowledge and analyzing the evolving distribution of roles between human capacities and the epistemic instruments that we use as surrogates.

Second, a conceptual framework is needed to investigate the scope of this epistemological inquiry about the epistemological novelty of simulations and their validation. General epistemology analyzes issues such as the nature, sources, or structure of knowledge and justified belief. Specific, applied epistemological inquiries can be pursued about the specific practices or processes through which beliefs are acquired within fields for which the promotion of epistemic objectives is important, such as adjudication, library science, journalism, or science. For example, the epistemology of science investigates how novel scientific contents are unraveled by processes such as mental reasoning, calculus, thought-experiments, experiments, or computer simulations (El Skaf and Imbert 2013).

A consequentialist framework like the one used by Goldman for social epistemology (Goldman 1999, 87) provides a useful tool to analyze the various aspects of such belief-generating processes. The inquiry is pursued relative to some epistemic states, such as knowledge, error, ignorance, or consensus, which are considered to have primary or fundamental value. Then, practices can be described as having instrumental value depending on how much they promote or impede the development of such epistemic states. Further, it is useful to adopt a fine-grained description of the resulting epistemic states. Goldman proposes the notion of "mental infosphere" at a time *t*, which consists of the beliefs of all the people inhabiting the globe at *t* (Goldman 1999, 161). Then, this conceptual framework is used "to widen epistemology's vista" (ibidem, preface) and to show how particular communication systems, adjudication rules, media funding systems, testimonial rules, etc., have a positive or negative impact on the dissemination of false or true beliefs in the mental infosphere. Similarly, one can analyze how much the practices involved in belief-generating processes that comprise computer simulations favor the development of true, error-free, consensual, reliable, etc., beliefs in the mental infosphere, or in its scientific part.

Belief-generating processes involving computer simulations require the expertise of specific scientists, organized in specific ways, dedicated to specific practices, and using specific tools, languages, or types of resource. Thus, trivially, their epistemological appraisal corresponds to a task in its own right since it requires analyzing and understanding new types of objects and processes. Further, because inquiries involving computational methods represent a large part of scientific activities, this task is an important one in scientific epistemology.

However, the importance and specificity of this *task* do not imply that, once epistemologists tackle it, they are always faced with novel *problems*. Nevertheless, could it really be the case that the computational, logical, mathematical, cognitive, and social specificities of computer simulations do not make a single epistemological difference and that all the epistemological problems that they raise boil down to problems that epistemologists have already solved in different contexts? If this is so, applied epistemologists should celebrate this cosmic coincidence and rejoice that their past works have such unintended scope.

In any case, even if simulations reveal epistemological problems that are similar or identical to those raised by other scientific activities, they can still be epistemologically different. Indeed, answering a traditional question about a novel object is not usually trivial. When one tries to solve equations of a new type, one tackles a novel problem. Saying that a mathematician who has successfully done so has achieved nothing new, because this is (once again!) the same old stew or problem of solving an equation would be mathematically naive. Naturally, it may be that answers to epistemological questions about simulations are sometimes identical to answers to similar questions about other activities, but it cannot be *assumed* that this will be systematically so. Finally, showing that a problem about some type of object is actually the same as another problem about another type of object is usually not simple; and showing that a problem reduces to another, or that the solution of the latter can be adapted to solve the former, is usually an achievement.

Overall, it is difficult to tell in advance how much the epistemology of computer simulation shares with that of other activities. Computer simulations have the same target as other scientific activities, rely partly on the same theoretical material, use common parts of applied mathematics, and are partly carried out by agents that are subject to similar cognitive, scientific or social constraints. Like experiments, they can be part of "big science", often involve using material and nonmaterial instruments, massive budgets, various collaborators, and may yield big data. Because all such features are not epistemologically neutral, the epistemology of computer simulations cannot be radically new, nor should it be carried out separately from epistemological inquiries about instruments, mathematics, computer science, statistics, experiments, and, naturally, scientific representations and models (see also Frigg and Reiss 2009, 611, Humphreys 2009, 615). In brief, there is no doubt that the epistemology of simulations has a lot in common with that of other scientific activities. Identifying genuine overlaps, disentangling and explaining shallow similarities from deep ones, and determining what is epistemologically specific to computer simulations strictly speaking and what is a general feature of computational science corresponds to a research program for applied epistemologists (for a critical overview of the case of simulations and experiments, see e.g., Imbert 2017, 34.5) In any case, claiming from the start that nothing novel is to be found in the epistemology of simulations remains puzzling. It is as if Columbus, after only one month in America, had claimed that there was nothing specifically new or interesting on this continent because local indwellers also had two legs and no road panel pointing at hot discoveries was in sight for newcomers.

For the following discussion, I adopt the following characterization of computer simulations (see Imbert 2017, 34.2.1 for more details):

> A computer simulation corresponds to the actual use of a computer to unfold the behavior of a physical system S, by generating a description of a potential trajectory of S in the state space of a computational model of this system by applying repeatedly an algorithm that computes the description of the next state of the trajectory from the description of the previous states.

Analyzing how the conclusions that are reached with the help of computer simulations can be validated requires discussing more than computer simulations per se. Computer simulations are embedded within larger scientific inquiries aimed to answer specific questions about certain target systems (El Skaf and Imbert 2013, 3454, Frigg and Reiss 2009, 596). At the end of the day, the key issue is not whether computer simulations faithfully represent some target systems but whether the data that they yield can be used to provide target questions with answers that are likely to be correct. Below, I shall consider that validation describes the process of making sure that this is the case. Validation in this sense is directed at inquiries and investigates the soundness of the production and use of computational results. As for any form of reasoning, the value of the final results hinges on both the content of the material that feeds them (typically premises, theories, or models) and whether the inferential process (which here includes the running of the simulation) is properly carried out. As we shall see, the process of validating simulation-based inquiries goes beyond the adoption of good scientific methods and has a social or communal dimension.

43.3 Dimensions of Computational Inquiries, or Where Things Can Go Wrong Epistemically

Belief-generating processes relying on simulation-based inquiries are extremely complex, from the elaboration and running of computer simulations to the reception of results in scientific communities and beyond. An important task for epistemologists is to pin down within these processes the various problems that scientists must solve for the final results to be valid, and where detrimental effects can be triggered and spoil the process.

Philosophers of science do not have a strong tradition of investigating scientific or epistemic failure. Inquiries about errors and practices are mostly carried out by sociologists of science (e.g., see the symmetry thesis about the explanation of false and true results in Bloor 1976), philosophers with specific orientations, such as naturalists, pragmatists, or advocates of a practice turn (see e.g., Kitcher 1993, Wimsatt 2007, Woods 2013), or philosophers investigating issues for which the question of errors can hardly be discarded, such as investigations about ampliative reasoning or statistics. Still, it is informational to locate within belief-generating processes the key factors (or "process variables") that influence the validity of results or on which the epistemic impact of the results hinges, if only to better control these epistemic processes. Another task is to analyze specifically the epistemic impact of the adoption at these hinge points of particular choices, behaviors, practices, or policies. When the corresponding epistemic effects are specific to the context of computer-based inquiries, or when such key factors are specific to such inquiries, the validation of simulation-based inquiries is a novel problem, which calls for specific epistemological analyses. I illustrate in the following paragraphs such cases.

43.3.1 The Production of Computational Results: Can We Control the Beast?

Because computer simulations involve carrying out a wide variety of tasks, they can fail in many, often specific, ways. Errors may come from the hardware, e.g., if single-bit alterations are caused by physical interferences. Failures may be rooted in the types of miscomputations or malfunctions that can affect digital computers and communication systems (Fresco and Primiero 2013), in particular in the written code or the software, which do not always do what we believe they do. Problems may come from the type of algorithms that we use to compute functions, to approximate real functions, or to solve equations, but also from their implementation; from the discretization of mathematical objects to make them amenable to computable descriptions; from a mismatch between the algorithms (or the models) and the type of computational architecture that is used (supercomputers now use parallel architectures, which require various adaptations), etc.

I do not presume to present here an exhaustive list; quite the contrary. Inventorying and analyzing specific ways of failing, from the hardware to the inquiry level, and assessing the factors that favor or neutralize them is a substantial epistemological task. Potentially, it requires a wealth of distinct expertise concerning various parts of the process, which clearly makes the situation epistemically uncomfortable for epistemologists. Still, this is no reason to discard or ignore this task, which is no less important than the analysis of potential sources of errors in other belief-generating processes, such as cognitive biases and fallacies of reasoning, logical errors in formal inferences, typical ways of failing in thought-experiments, or bandwagon effects and cascades in belief exchanges within communities. I highlight below aspects of the production of computational inquiries that make their validation specific.

43.3.1.1 Computational Practices: An Evolving Field with Specific Epistemic Values

The success of computational science depends first on the ability of scientists to develop specific technical, mathematical, and human strategies to solve hardware and software problems effectively (for questions pertaining to verification, see Sect. 43.3.2 below and Chap. 10 of this volume by Rider).

These questions cannot be discarded or identified by an armchair inquiry, since the issues of where reliability bottlenecks lie, how they are faced, and which questions are hot concerning computer simulations and their validation keep evolving with technological and scientific progress. For example, for the first computers, "the overwhelming problem was to get and keep the machine in working order," as is reflected in the names of societies such as the Association for Computing Machinery (Dijkstra 1972, 860). Also, the existence of single-bit alterations may no longer be a worry for ordinary simulations, even if it remains an issue for sensitive simulations, for which error-correcting code memory (ECC memory) devoted to scientific computing needs to be used. Similarly, in the late 1960s, the development of computer power had triggered needs that could not be answered by programmers' abilities. "Programming ha<d> become an equally gigantic problem" (ibidem, 860), and "the software crisis" had arisen, with the development of low quality, inefficient, or difficult to maintain software. How scientists have managed this crisis ever since is a question worth exploring.

Anyhow, beyond discussions about the validity of particular inquiries, the *average* validity and *global* impact of computational inquiries depend on various properties of hardware and software. How much hardware and software is globally *efficient, easily usable, standardized, maintainable, adaptable* for follow-up inquiries, *transferable* to other scientific problems, etc., influences how much sound results are produced. These properties correspond to epistemic values that are specific to computational inquiries, so investigations about their impact are clearly novel. For example, a science in which all codes are radically different and all practitioners develop their specific solutions is unlikely to be efficient and globally reliable. In contrast, the existence of shared codes of good practices and commonly developed software tools, or the adoption of common standards (e.g., in terms of hardware, programming languages, or mathematical tools) is bound to have positive effects. This shows that the epistemology of computational inquiries, like that of instruments, goes beyond that of individual practices and overlaps with social epistemology. How much the above properties need to be traded against one another and where actual computational practices within empirical science lie on this multidimensional map are other questions worth investigating.

43.3.1.2 Applied and Computational Mathematics for Limited Social Agents: The Case of Random Numbers

To carry out computer simulations, scientists need to find ways of solving various mathematical problems. Understanding how they do so requires going beyond traditional questions in the epistemology of mathematics such as how we interact with mathematical entities, make reference to them, or access mathematical truths, since the epistemological issue of how we *develop* mathematical knowledge remains largely untouched by answers to these foundational questions. By contrast, the epistemology of applied mathematics and computational science deals directly with the issue of how logical and mathematical content is *unfolded*, knowledge *extracted*, and problems *solved* given our limited wherewithal, the complexity of the task, and the features of the formal tools that we use (see e.g., Wimsatt 2007, El Skaf and Imbert 2013, Fillion and Corless 2014, Lenhard and Carrier 2017). Accordingly, it involves analyzing how heuristics for mathematical problems work and what we can expect from them; how to describe the quality of approximate solutions, how to develop mathematical strategies to analyze and control computational errors and, more generally, which features influence how applied mathematicians crawl their way through complex problems. While applied mathematics is not limited to its use in computational science, it is central to this field, and much of it is developed for the needs of computational inquiries.

For illustrative purposes, I now present the case of the production of random numbers by simulation practitioners and highlight factors on which the reliability of this task depends. The production of random numbers is a central problem of modern science. Randomness is a key concept across various theories, and fields and scientific arguments involving statements about random properties are frequent. A specificity of computer simulations is that they often rely on the use of token numbers that *instantiate* the property of randomness (*versus* involve statements that attribute it). Accordingly, the validity of statements *about* random properties merely relies on the semantic relation between the content of these statements and what they denote. By contrast, that of computer simulations and computational inquiries using random numbers also relies on our ability to produce such random numbers. Various epistemological questions arise in this context. How easy is it to produce the random numbers that our computer simulations need? Which factors have an impact on this production? Can we expect the random numbers that are usually used in computer simulations to be good enough for the preservation of the validity of inquiries? I provide evidence that these questions cannot be ignored and have nontrivial answers that require going into the details of socio-computational practices, and perhaps the psychology of practitioners.

Whereas almost all sequences of binary digits are random, producing random numbers is extremely difficult. The need for scientists to produce many such numbers increases the difficulty of the task. Random number generators (hereafter RNG) must satisfy various requirements such as producing uniformly distributed, reproducible, random numbers, which have periods that are much larger than the samples used. Jointly fulfilling all these requirements is in general difficult, but even more so in the context of parallel computers (Hellekalek 1998). Parallel architectures involving thousands of processors were developed in the 1990s and supercomputers are now massively parallel. Then, to the extent that parallelization is possible, parts of the computational task can be computed synchronically, which speeds up computation. Typically, replicas for Monte Carlo simulations can be produced independently. Thus, good random generators should be parallelizable. For parallel RNG (hereafter PRNG), other requirements are the absence of correlations and, for reasons of efficiency, the need to generate numbers independently (ibidem, 85). After reviewing existing methods to fulfill these requirements, Hellekalek concluded that it was "not at all trivial to find high-quality RNGS for parallel machines" (ibidem, 82) and that some aspects of the problem (e.g., correlations between disjoint substreams of consecutive numbers) were "dangerous territory" (ibidem, 85). Indeed, his analysis showed that the application of parallelization techniques to standard RNG could "perform terribly" (ibidem, 86). Thus, his paper was named "Don't Trust Parallel Monte Carlo"—arguably a big stone in scientists' shoes, given the importance of parallel computers and Monte Carlo methods for computer simulations.

Naturally, things have improved since Hellekalek's paper, but full optimism may still be inadequate. Scientists' needs have also increased massively and access to supercomputers is difficult. Simulations in nuclear medicine can require as many as 10^{20} random numbers for computations carried out on thousands of processors. A fundamental problem is that scientists do not have theorems or techniques to prove the independence of two parallel random streams (Hill 2015, 68). Further, strong autocorrelations within pseudorandom numbers can appear far apart and spoil the application of parallelization techniques (De Matteis and Pagnutti 1988). In practice, the testing of PRNG is based on a battery of statistical tests, such as BigCrush TestU01, which represent the current state of knowledge about random numbers. and few PRNG satisfactorily pass the test. The epistemological moral is that it can require pointed expertise to determine whether a (P)RNG is sufficient for a scientific inquiry.

The next question for epistemologists is to assess whether the (P)RNG that are *actually* used in science are in general satisfactory. After all, if scientists always use the currently best RNG, troubles are unlikely, and epistemologists should not bother. Evidence can be found that epistemological optimism may be misplaced again. For a scientist without expertise about RNG, the easiest option is to use the "rand" functions from standard libraries that are used by her community. The problem is that "almost all of these generators are badly flawed" (Jones 2010), and the somewhat inconvenient advice here is to "always use <one's > own random number generator" (ibidem). The use of dedicated libraries is no guarantee either. In 2004, Joel Heinrich, a researcher in high-energy physics, pointed out serious defects in pseudorandom generators provided by standard libraries such as Linux C and C++ as well as a major bug in CLHEP (*A Class Library for High Energy Physics*) class library for random generators, that is, tools frequently used by physicists (Heinrich 2004). Similarly, 40 out of 58 generators in the GNU scientific library were shown to have defects due to inadequate initialization schemes (Matsumoto et al. 2007). Indeed, an inadequate use of a good RNG can also spoil the broth. For example, a simple way to seed an RNG is to use the time function existing in most libraries. However, this is not acceptable if one launches too many jobs because a large number will start at the same time on different nodes. This leads to a repetition of the same calculations many times and surreptitiously spoils the statistics (Jones 2010). Unfortunately, this type of problem can be hard for practitioners and for the external community to detect. Overall, for random number production, the use of communal scientific resources does not protect against failure. Thus, because many scientists lack the relevant, evolving expertise and do not always resort to experts for such local choices, computational results may often be sullied.

Even then, reliability may be preserved by the transmission of good computing practices within communities (see e.g., Wilson et al. 2014). Thus, the average validity of computer simulations using random numbers depends on whether communities are organized in a way that favors the adoption of sound practices and indirectly promotes reliability. Social epistemologists of science should then investigate whether the right information is easily accessible and actors are incented to do the right thing. Good practice guides like that of Jones (Jones 2010) can help practitioners adopt appropriate practices or understand that they need assistance. Explicit publication standards in good journals can also point out sensitive aspects, if, for example, authors are systematically requested to provide details about the nature, properties, and implementation of the RNG that they have used. Overall, on this and other issues, the reliability of computational science is contingent on the adoption of appropriate practices at the community level and on how individual scientists tend to behave in this unsafe environment.

43.3.1.3 Changes in Modeling Practices, Justification Strategies, and Typical Usages

Based on their familiarity with particular types of simulations, some philosophers have tried to extrapolate and to single out specific features of the epistemology of simulations. For example, in early writings, cellular automata were seen as typical illustrations of the epistemological novelties brought about by computer simulations. More recently, Winsberg has suggested that the knowledge produced by simulations results from inferences that are *downward* (from theories to phenomena), *motley* (the justification process is a combination of disparate elements), and *autonomous* (see, e.g., Winsberg 2010, passim).

Unsurprisingly, such claims are easy to falsify. Simulations are versatile, mostly neutral inferential tools. Like other general tools, they can be used in various (epistemic) contexts and depending on the cases, the appropriateness of their use can be justified in different ways. In brief, simulations are epistemologically heterogeneous and the project of finding some general, novel features about how their results are justified seems doomed to fail. Because models are also versatile tools, their epistemological uses are also heterogeneous. At the end of the day, it is no surprise that epistemological features instantiated by simulation-based inquiries can also be instantiated by (pen-and-pencil) model-based inquiries. The conclusion should not be that the general epistemology of simulations boils down to that of models but rather that neither (the class of) model-based inquiries nor (the class of) simulation-based inquiries correspond to epistemological kinds that provide appropriate units of analysis for general investigations about validation strategies.

To discuss validation strategies and assess the different types of roles that simulations can play within these strategies, inquiries should be described at a fine-grained level by specifying their goals, what is known about the target systems, what tools and resources are available, etc. Then, it may be the case that simulations sometimes open up a space for novel validation strategies or, more frequently, for new versions of existing strategies. Generic modeling practices such as approximating, idealizing or abstracting are a way to use models which, though tractable and simple, produce results that suit the particular goals of inquiries. While these procedures are already analyzed in the literature on scientific models, it remains worth investigating whether these generic practices have specific versions and require particular epistemological scrutiny in the context of simulation-based inquiries. Similarly, ensemble forecasting can be seen as nothing novel since it amounts to combining different incompatible epistemic sources (here simulations) to make (predictive) judgments. However, it is usually agreed that this procedure calls for specific analyses in the context of simulations and climate analysis.

Importantly, the epistemology of science should analyze what validation strategies are used in suitably described contexts, but also how frequently these strategies are used, and why. Epistemological features like those highlighted by Winsberg, though not specific to simulations, may correspond to strategies that develop with computational science. Arguably, because computer simulations are a powerful tool, they are likely to be used in more complex and uncertain cases, which would not be investigated otherwise and which constrains the selected strategies. Then, *because of this type of use*, computational inquiries may seem, *on average*, to have specific features and an epistemology of their own and they may modify how science is usually practiced. For example, simulations may more often involve approximations and departures from the truth, epistemologically mixed or impure methods, trade-offs between epistemic goals, etc., even if, when philosophers of science analyze their aspects individually, the practices that they find are not radically different from those identified for other types of model-based inquiries (see Imbert 2017 for more details and similar analyses about unexplanatoriness and simulations). Overall, the failure to note the difference between analyses in terms of properties of token inquiries and analyses in terms of frequent features of inquiries may be another cause of dissent concerning the novelty of the epistemology of simulations.

In any case, computational science may require a specific analysis with respect to typical modeling or justification practices within communities. For example, the availability of computational power may change which modeling strategies are most often used. As noted by Frigg and Hartmann (2017, Sect. 3.1), computational power may encourage scientists "to swiftly come up with increasingly complex models." This may lead in turn to an improvement of the empirical adequacy of predictions, but not necessarily to a better understanding of underlying mechanisms. In the end, such changes may modify which goals are valued and which modeling norms are dominant within the cultures of communities using simulations. Differences of these types can hardly be analyzed by scrutinizing exclusively the content of particular representations.

43.3.1.4 Division of Scientific Labor, Computational Inquiries, and the Preservation of Validity

Various types of tasks requiring different types of expert knowledge need to be adequately carried out for the production of valid computational results. Because no single individual can possess all the relevant knowledge, scientists need to divide the global task into subtasks, and delegate their completion to specific humans or machines. Then, how much computational science can felicitously "<push> back the boundaries of what can be known" (Humphreys 2004, 154), depends on how much safe practices of dividing labor, which does not compromise the validity of the global inquiry, can be applied. The possibility to divide inquiries into standardized nontrivial units or modules that can be carried out independently and recombined together to yield sound results is beneficial, in particular for the validation of simulations. For example, different actors with pointed expertise can be in charge of each module and produce more reliable collaborative inquiries; some failures can be more easily localized by means of local tests; other failures may have local impacts, etc. The advantages of modularity are not specific to computer simulations. However, how much modularity is possible and beneficial for simulations and their validation requires a specific investigation. The following argument can be used to clarify the situation and explain why validating simulations can be difficult.

P1 If, in a perfectly modular structure, each individual module works, so does the whole structure.

P2 Simulation-based inquiries are perfectly modular.

P3 It is straightforward to check whether the modules of simulation-based inquiries work.

∴ It is straightforward to check whether simulation-based inquiries work

Evidence seems to suggest that conditions P2 and P3 are often false for simulations. For example, while well-designed modularity is desirable, it is often conspicuous by its absence from programs: "patches, ad hoc constructions, bandaids and tourniquets, bells and whistles, glue, spit and polish, signature code, blood-sweat-and-tears, and, of course, the kitchen sink—the colorful jargon of the practicing programmer seems to be saying something about the nature of the structures he works with" (Millo et al. 1979, 277). In various cases, there is uncertainty about how much modules actually work and whether potential departures from exactness are a worry. Typically, mathematical functions are often approximately computed by the versions that libraries provide. If users are not strongly aware of the limits of the specific functions within libraries, the results can be corrupted (see the case of random numbers described above). Lenhard also supplies the example of the practice of "kludging", i.e., using quick-and-dirty and hard to maintain solutions to make software work (Lenhard, forthcoming, see also Chap. 38 by Lenhard in this volume). This implies that knowledge concerning the validity domain of parts of the software can become lost. Further, in many cases, the fact that black-boxes are used makes it impossible to check deeper into the modules. Overall, this means that the global validity of computational inquiries can become corrupted and uncertainty often remains as to whether this is the case. In practice, modularity may be a solution, but not always a blissful one.

The various reasons for this corruption of modularity, whether it is inevitable, and the strategies developed by practitioners to preserve validity when modularity is eroded, are other questions worthy of study. Answers may differ depending on the aspects or fields considered. For software architecture, there are clearly "reasons for degeneration: ongoing evolutionary pressure, piecemeal growth. Even systems with well-defined architectures are prone to structural erosion" (Foote and Yoder 1999, Chap. 29, quoted by Lenhard). At the same time, "a sustained commitment to refactoring can keep a system from subsiding into a big ball of mud" (ibidem). How much safe modularity is preserved depends on which types of tools, practices, and norms are actually adopted within a scientific community, from the hardware to the modeling level. This is again a contingent issue, which epistemologists cannot analyze by armchair analyses. Importantly, different factors pull in different directions. Modularity brings about epistemic advantages, such as the facilitation of piece-wise validation and understanding of inquiries and their results. However, preserving modularity can be extremely costly. Similarly, reusing and adapting modules beyond their initial domain of validity to produce more results quickly is a legitimate concern, even if this tends to make errors more likely. Describing more precisely the trade-offs between these different epistemic goals can help to understand the constrained epistemic choices that resource-limited practitioners and communities are faced with, why some practices are considered as good, acceptable or sloppy, or why some types of errors or problems can be expected within computational inquiries. Depending on the orientations that are taken by communities concerning these matters, different types of computational science are possible.

43.3.1.5 Computer Simulations for All: What Epistemological Effects?

Computational science increasingly benefits from the development of various tools at the hardware, software, or modeling levels. Individual scientists would not be able to complete many inquiries without all these computational, mathematical, and modeling facilities. This situation keeps lowering the epistemic cost of the *running* of computer simulations (in terms of what one needs to know). Even scholars within communities with no strong training in computer science and mathematics can develop potentially valid simulations. But is this really safe: can individual scientists *really* afford epistemic ignorance and still produce sound computational results, or is this a lure? Actually, the epistemic price to *validate* results properly may remain high.[1] Indeed, the question of whether partly reliable tools and facilities work well usually calls for a context-specific answer, and determining this answer requires expert knowledge concerning both the tools and the subject matter. Thus, new tensions arise from these modern facilities, which offer opportunities to produce a wealth of results across scientific fields but come with new risks of failure. While this tension exists for other complex activities, it is extremely acute here. How scientists eventually behave, i.e., how much they cope on their own or ask other experts or collaborators for help hinges on many factors. These include the cost of human and computational resources, how much failure is risky and acceptable, whether errors are often detected by peers and tarnish scientific reputation, etc. The productivity and reliability of computational science can vary significantly depending on what is the case concerning such factors.

43.3.2 The Reception and Post Hoc Assessment of Computational Results

A bad result that is used has a detrimental impact. A sound result that is ignored has no beneficial effect. In both cases, we are epistemically worse off. Accordingly, epistemology must also scrutinize how results are publicly validated, accessed, trusted, and used once they are produced. I highlight below a couple of issues that make this problem specific for computational inquiries.

43.3.2.1 Epistemic Access

For mind-produced results, inferential processes and their conclusions, *qua linguistic entities*, are accessible to the authors, who personally carry out these activities. Publication extends this access to the public. Things are different for computer simulations. These are carried out by external processes. Thus, the authors no longer have a privileged epistemic position. Furthermore, even if the content of computer simulations can be described logically (putting aside issues concerning physical implementation) and can be made accessible provided that scientists preserve bit-reproducibility (Demmel and Nguyen 2013), in practice, practitioners usually cannot access the details of computational processes. In other words, simulations remain *globally epistemically opaque* (Humphreys 2009), even when they are *locally transparent* (Imbert 2017, 726): a human mind can inspect any part of the process though it cannot inspect all the parts. Things are worse for the more distant scientific audience. In most cases, the public can access a tiny fraction of the results through tables or graphs. In some cases, the whole data set and the code are available for inspection, while in rarer cases, this is true for the whole state-by-state simulation. However, it

[1] Similarly, knowing the main effects of drugs may give lay people the illusion that they can safely decide whether they should take them when they are sick.

is virtually never so for the bitwise computational process. Overall, how much of the process can be accessed, directly (by human minds) or indirectly (with software facilities), depends on computational questions, publication policies, issues related to openness and proprietary use, or the development and maintenance of storage facilities and exploitation software. (Note that the overlap with similar issues for experimental science is merely partial). Since public validation and good use are contingent on the possibility of access, the epistemic impact of simulations clearly depends on how this problem is socially and technologically treated within scientific communities.

43.3.2.2 Verification of Program Correctness

Accessing results is one thing, trusting them and using them is another. It does not matter that some type of process often produces adulterated results if its users can identify and use sound cases. In brief, whether it is possible to certify the reliability of simulations is crucial to their felicitous use. Here again, the epistemology of simulations overlaps with that of other activities but it has its specificities.

At a low level, program verification is a matter of verifying whether token computational runs have the appropriate causal behavior, which is a specific version of the problem of inductive inference (Fetzer 1988, passim). At a higher level, it can be seen as that of verifying whether algorithms and their coded counterparts do what they should. De Millo et al. (1979) argued that program verification does not work like proof verification. Mathematical proofs are usually sketches of proofs (*versus* formal proofs in the logician sense), and their logical validity is publicly discussed by mathematical communities. Program verification is different, because proofs of program correctness for real-life systems are long, tedious, and repetitive, and are not usually published nor publicly discussed (De Millo et al. 1979, 276). Dijkstra (a defender of program verification) counterargues that proofs of program correctness can also be the object of lively exchanges between scientists. Further, trivial mathematical theories also have simple statements "whose finite proofs are impossibly long" (Dijkstra 1978). Thus, for both proofs and programs, mathematicians need to find concise and elegant proofs.

Fortunately, for the purposes of this chapter, there is no need to endorse a position about the nature and ideals of program verification. Epistemology deals with what we can do *in practice* and what we actually do, given our epistemic, computational, and cognitive wherewithal and the incentives within our epistemic communities. Similarly, Wikipedia may change nothing of the nature of knowledge, justification, and science, but it changes how agents with limited resources and cognitive biases access knowledge. Thus, its existence modifies our epistemological situation and changes which beliefs propagate throughout human societies. Here, even if proofs of program correctness and mathematical proofs share the same nature and ideals, in practice, strong epistemological differences between them remain. If proofs of program correctness are usually not published, are less valued as scientific achievements, less scrutinized, much longer and extremely repetitive, then, from an epistemological point of view, program verification works differently.

Further, verification of programs is de facto a partly specific problem for scientific activities outside mathematics and computer science. Millions of lines of code are regularly written for the purpose of scientific activities. The more software facilities develop, the more scientists with no specific background in computer science write code, which is neither verified by formal methods nor undergo the interested scrutiny of computer scientists or mathematicians. This raises the question of how these codes are actually tested and how reliable related procedures are. In the absence of a grand theory of testing, "programmers are probably better off using the tools and insights they have in great abundance. Instead of guessing at deeply rooted sources of error, they should use their specialized knowledge about the most likely sources of error" (De Millo 1978, 41) and rely on their "intuition and problem-dependent knowledge in a disciplined manner to test for a variety of specified error types" (Shapiro 1997, 31). Further, program correctness merely guarantees that the implementation matches the specifications; but these can themselves be flawed (Shapiro 1997, 32), and unexpected physical, mathematical or computational conditions or situations can bring about failure. In brief, testing often relies on a messy combination of formal and nonformal, subject-specific, and partly dirty strategies. Thus, while the epistemology of computer simulations and software engineering is at the crossroads of other disciplines and overlaps with them, it does not reduce to them and requires a specific scrutiny from philosophers of the empirical science, even if they still lack the corresponding culture.

43.3.2.3 Verification of Mathematical Correctness

Even if computer simulations work properly at the hardware and software level, they may be unsatisfactory because they compute solutions that are not close enough to the unknown solutions of the target models or equations. In the frequent absence of mathematical theorems to guarantee that this is so, assessing whether this is the case is difficult. I will not develop this point here, as it has already been discussed in the literature (see e.g., Winsberg 2010, Chap. 2, Frigg and Reiss 2009, 603).

43.3.2.4 Reproducibility

Scientists can be willing to replicate or reproduce simulations and their results. Replication is costly and not all scientific results or simulations are replicated. Nevertheless, the possibility of replication is a cornerstone of science and the validation of scientific results. In principle, it is possible for entities that can be defined or presented unambiguously by linguistic means. By contrast, replicating experiments or thought-experiments can be controversial, which may feed epistemological problems like that of the experimenter's regress (Collins 1985).

Over the past decade, there has been a growing awareness that present scientific practices and publication rules often do not match replicability standards (Baker

2016). This is referred to as the replication, replicability, or reproducibility crisis in science. Until recently, it was seen as touching almost exclusively experimental science. As it turns out, computational activities are also concerned. Failure to reproduce computational results or to replicate a computation can stem from various sources: the authors may not share their code; the representation of real numbers may vary; the order of associative operations such as addition and multiplication may make a difference in floating point representations; programming languages, compilers, operating systems, and finally computational architectures may make a difference (Hill 2015), etc. How serious is the problem for computer science and computer simulations in particular? Some researchers like Claerbout have struggled over the years to create a reproducible research environment and have reported how difficult this has been (Fomel and Claerbout 2009). More recently, Collberg and Proebsting tried to replicate computer science research presented in 601 papers from the respectable Association for Machinery conferences and journals (Collberg and Proebsting 2016). They defined different degrees of repeatability based on how difficult they found it to repeat the research. In spite of their efforts, 47% of the 601 target papers turned out to present non-repeatable research. It is unlikely that computer scientists are more careless concerning reproducibility than researchers who simply use computer simulations for their research. Accordingly, epistemologists should not indulge in wishful thinking concerning the replicability of computer simulations, and, arguably, computer simulations also raise a specific reproducibility problem.

43.3.2.5 Trust

Overall, the impact of computational results depends on how much and when the results of computer simulations are trusted, and whether this trust is misplaced or not. If one takes a general, abstract, bird's eye view of this problem, it looks familiar and seems to boil down to the issue of how and when scientists accept being epistemically dependent on their peers and using their results (Hardwig 1985). The answer can be described in terms of networks of trust or trust indicators such as the scientific reputation of journals, scientists, or institutions. However, at a more fine-grained level, how much trust toward computational results is distributed and how these trust indicators are fed depends on the details of practices across fields. Here again, computer simulations may require specific scrutiny.

43.3.2.6 Publication Procedures and the Setting of Appropriate Standards

Publication procedures contribute to the production, assessment, and diffusion of good results. Tuning them appropriately for computational inquiries can be specifically beneficial with respect to some of the problems described above. I shall give brief examples here.

Because access to relevant information is crucial for replication or validation by peers, but also for novel inquiries that use existing data (e.g., those generated by big simulations such as the Millenium Run), editorial rules, or requests concerning what information authors must provide, as well as openness and proprietary issues, can influence the epistemic impact of computer simulations.

Editorial rules can also be used to keep "educating" members of scientific communities about what can spoil simulations (e.g., if authors are requested to provide detailed information about the (P)RNG they use). This is particularly true since computational science evolves at a brisk pace and communal practices need to keep adapting to guide individuals.

Beneficial results can also be achieved through appropriate authorship practices. Collaborative science is now widespread, which may undermine epistemic accountability and feed a decrease of reliability (Andersen 2014, Imbert 2014). In this context, major journals, like *Nature* or *JAMA*, have started adopting policies to make authors list their respective contributions, and what they endorse responsibility for (Rennie 2001), as well as who the guarantors are (Rennie 1997). In the context of computer simulations, adapted versions of these policies may be adapted to indicate the crucial scientific roles that must be endorsed to carry out and validate simulations properly. This may put virtuous pressure on practitioners, e.g., concerning the interpretation of agent-based models by computer scientists with no object-specific expertise, or the internal validity of simulations carried out by researchers with little expertise in computational methods.

43.4 Should Epistemologists of Science Bother, After All?

As seen above, belief-generating processes involving computer simulations can fail in various places, spoiling their epistemic impact. Can epistemologists ignore these issues? Epistemologists of science have a strong tradition of focusing on scientific representations. So far, the issue of the validation of simulations has mostly been tackled through the lens of the epistemology of models and question such as whether partial misrepresentations (e.g., due to idealizations, approximations or abstractions) threaten the conclusions that can be drawn from models (Frigg and Hartmann 2017). I now provide general arguments to the effect that an adequate epistemological analysis of computer simulations should extend beyond these questions.

43.4.1 Target Models, Actually Investigated Models, and Failure

When a simulation is carried out, a computational model is always exactly explored, even if it differs from the model targeted for investigation. So the validity of a simulation always boils down to that of this computed model. Therefore, why extending investigations beyond those of models?

Unfortunately, such a position begs the question. First, the model that is actually computed can be unknown, e.g., because unnoted errors spoil the investigation, so the above position makes scientific failure more difficult to analyze. Second, the exact description of these actually computed models should include hosts of gory mathematical and computational details as well as information about the software and hardware, i.e., much more than philosophers analyzing models usually discuss. Thus, at the very least, one must distinguish between the target models that one would like to investigate and those that are actually investigated, knowingly or not. Third, focusing on the content of target models also remains unsatisfactory. From an epistemological point of view, what matters is less the potential of target models than what is actually extracted from them by practitioners. Typically, if some Monte Carlo practitioners use low-quality random numbers, their results may be incorrect, whether or not the target average quantity in the model represents the target system property correctly. Overall, analyses about models and mathematical–computational practices are complementary. Just as investigations about the death toll on roads cannot be reduced to analyses of the driving code and road maps, discussions about the epistemology of models and scientific representations cannot save us the effort of epistemological investigations about mathematical–computational practices.

43.4.2 The Valuable Redundancy Argument

Let us assume for the sake of the argument that the epistemological analysis of computational models could exhaust that of computer simulations. Even so, much independent epistemological work would be needed to describe how other aspects of simulations, such as coding, mathematical practices, verification procedures, etc., favor the production of reliable results. This can be understood with an analogy to classical mechanics. Even if one knows exactly the trajectory of a deterministic system, there remain hosts of regularities to be discovered between other variables describing the system. These regularities are somewhat redundant, since anything about the system's behavior can be derived from the knowledge of its trajectory. Nevertheless, discovering such regularities remains epistemically valuable. Similarly, investigations into the reliability of computational practices and their epistemic impact are valuable, even if the validity of computer simulations is determined by the very content of the models that are investigated.

43.4.3 The Procrustean Objection

Finally, one might argue that many of the questions described above are novel but belong to formal or empirical science. As such, they might be discarded from epistemology, leaving nothing substantially novel in the epistemology of simulations. For example, Frigg and Reiss emphasize that questions, e.g., about the relationships between numerical and actual solutions or the impact of truncation errors are "purely mathematical problems" (Frigg and Reiss 2009, 592, 602).

This type of answer is perplexing. Once one has provided clear-cut and consensual notions of what counts as logical, mathematical, epistemological, etc., inquiries, one can analyze which questions fall within the scope of these inquiries. This is what I have done above with the consequentialist epistemological framework used by Goldman for social epistemology. In this perspective, nothing precludes that some problems or sub-problems belong to several disciplines—or one should explain why, whereas disciplines are historical and partly conventional constructs, there cannot be a partial overlap between them. The sub-problems that need to be tackled to pursue epistemological inquiries can also be considered to be epistemological ones, although perhaps derivatively. It would be implausible to claim that (the solutions of) philosophical or epistemological questions cannot involve (those of) mathematical or scientific questions.

Let us take an example. Suppose that one pursues epistemological investigations about journalistic practices and how much they promote the diffusion of true beliefs (see e.g., Goldman 1999, Chap. 6). Then, the solutions of various cognitive, technological, sociological, or economical questions about journalism and communication systems are relevant to these investigations and overlap with them. Further, these investigations coincide with those pursued by "theoretical journalists", who search for demonstrably reliable journalistic practices. However, epistemologists of journalism do not assess either the reliability of particular pieces of information or whether journalistic rules are applied correctly.

The case of science is analogous. Scientists try to develop safe practices to extend scientific knowledge. Thus, proving results about the reliability of particular methods, applying sound practices, and assessing the validity of particular inquiries is directly their task, even if it may provide indirectly relevant information for epistemological inquiries. Epistemologists analyze science and the reliability of its practices in order to present a faithful picture of science, given the epistemic, technological, sociological, etc., conditions in which it is practiced. Then, the assessment of the general reliability of scientific practices or possibility or impossibility results about these practices is common concerns for both inquiries. Emphasizing this overlap does not amount to confusing the goals and tasks of scientists with those of epistemologists.

Overall, if it deals with the epistemic analysis of natural belief-generating processes, epistemology inevitably intersects the fields that investigate specifically these processes, what they are and what they can be like. Accordingly, impossibility and complexity results in mathematics, logic, and computer science, results in cognitive and social psychology about reasoning and biases, results about the aggregation of individual judgments and preferences, or sociological analyses of how scientific communities work intersect epistemological inquiries. In the present case, mathematical questions about the complexity of verifying programs or psychological and sociological questions about how computational communities are organized epistemically and how scientists behave within them overlap with epistemological inquiries about the validation of computer simulations. Discarding the epistemological dimension of these intersectional questions for the purpose of a non-novelty argument amounts to elaborating an inadequate Procrustean version of epistemology.

43.4.4 The Absence of Data Argument and the Ostrich Strategy

Pointing at problems that can spoil the validity of computational inquiries is one thing, nevertheless, some may be already solved or may have a minor impact. Thus, one would need to know which of these problems frequently generate errors that threaten the validity of simulations, and which can be idealized away.

Unfortunately, it is extremely difficult to provide data about how often and why computer simulations fail since correct results are usually unknown and cannot be used as an external standard. However, the absence of accessible evidence about something in no way disproves its existence. There are many reasons why failures of computer simulations are not likely to be detected or publicized when they occur. First, simulations are not self-certifying activities in the sense that simulating a system does not produce direct evidence by itself that the simulation is successful. By contrast, juggling is self-certifying: when one juggles correctly, one immediately knows about it. Second, when computational inquiries unknowingly fail, usually some data are still produced. Once criteria of syntactic correctness are met, computer simulations always yield numbers, and practitioners need to deploy specific vigilance to track potential troubles. Third, not all robustness tests (e.g., by using different computational architectures, codes, libraries, etc.) can be carried out. Fourth, external detection of failure is often difficult because the details of computational activities are usually not public, replication is difficult, and incentives for replications are low. Fifth, because problems can be potentially ascribed to various tasks in the process, localizing failures means facing a specific version of the Duhem–Quine problem (Winsberg 2010, 24, Frigg and Reiss 2009, 604). This undermines scientific accountability and may encourage sloppier practices. Finally, even when failures are detected or suspected, nothing may happen, unless something major is at stake. Scientific life is short, resources are scarce, publicly localizing others' errors is time-consuming, and pay-offs for doing so are usually low. Accordingly, scientists may simply do nothing and let the results that seem fishy feed the gray zone of science. Overall, it is difficult to assess computational failure satisfactorily. Direct methods, e.g., by counting public detections of errors or retractions, are likely to grossly underestimate it and computational science runs the risk of the *invisibleness of its failures*.

It is often sound policy to leave aside issues that one cannot treat correctly. Nevertheless, the difficulty of directly observing some phenomena and the unavailability of objective standards for evaluation purposes are frequent in science, and they do not discourage scientists. The epistemological analysis of adjudication systems raises similar problems, because one never knows what the right verdict should be (Goldman 1999). Yet indirect ways out of the deadlock can be found out. For example, when a lay jury and a jury involving judges, or a real jury and a mock one give different verdicts, the two cannot be correct. This was used to investigate the effects of jury size and decision rules (Kalven and Zeisel 1966, Hastie et al. 1983). Here, the inability to replicate computer simulation results may, for example, be used as a general indicator of their invalidity. If some authors do not manage to replicate some computational results, this could mean that these results are sensitive to the method used, and their supposed scope is usurped. Alternatively, the method could have been badly implemented, or there may be some initial vagueness concerning the target model, which often surfaces when codes need to be effectively written. Overall, dropping the case of the epistemological assessment of computational practices on the ground of armchair arguments or, because it is difficult, would be tantamount to behaving like ostriches, which according to rumor bury their heads in the sand in the face of danger. Further, given the evidence that computational practices can fail in various specific ways, the burden of proof lies on the shoulders of the epistemologists of science who claim that sources of failure for simulations can be idealized away or ignored, except when it comes to their pet research topic (typically misrepresentation for philosophers of scientific models).

At the end of the day, I have no optimistic or pessimistic general conclusion to make about the present validity of computational inquiries. The point is rather that, given the *type* of activity that they are, and all the factors that can spoil them, it is not difficult to figure out states or domains of science in which simulations are sloppy or unreliable methods. Thus, it is worth investigating what is the case, why, and whether things can be improved epistemically.

43.5 Conclusion and Moral

Computer simulations have changed science. Over the past decades, it has been claimed that they also needed a novel, if not revolutionary, epistemology. Some such claims were over-stretched and the criticisms they raised were legitimate. However, one should be careful not to throw away the baby with the bathtub water. I have tried to present a sober version of the thesis of the epistemological novelty of simulations. I have adopted for clarification purposes a conceptual framework borrowed from Alvin Goldman and used it to emphasize that the computational, mathematical, representational, social, and potentially psychological dimensions of computational inquiries and their reception within scientific communities require specific epistemological investigations if one is to understand their validity and their epistemic impact. These investigations often raise novel questions, especially with respect to objects of novel types, like hardware and software, or call for novel and context-specific answers to traditional questions. I have not discussed the cognitive dimension of inquiries based on computer simulations, even if it is potentially an important one. For example, how we cognitively handle code or complex computational models, control computational activities, interact with computers, analyze and grasp computational data, which specific skills are required for these activities, and which type of biases are more frequent in this context are questions worth investigating. I have not discussed either how much the epistemology of activities like theorizing, predicting, evaluating, corroborating, explaining, or understanding, is altered when it is carried out by means of computer simulations (see Imbert 2017 for the last two questions).

Are these conclusions about the epistemological specificity of simulations so surprising? Over the past three decades, epistemologists and philosophers of science have provided analyses that Kitcher characterizes as belonging to the return of naturalists (Kitcher 1992). Against epistemological investigations that are almost exclusively centered on the content of representations, such approaches emphasize the epistemological importance of studying the various aspects of belief-generating processes, in particular, their psychological and social dimension (see Kitcher 1993, Solomon 1994, Goldman 1999, and Kitcher 2002 for an insightful overview). The above analyses fit within this naturalistic perspective and show the need to include a computational dimension, broadly construed, to epistemological analyses when computers are part and parcel of scientific belief-generating processes, which is an increasing majority of cases.

Overall, such a naturalistic epistemology is bound to be demanding for students of science. Philosophers of empirical science usually have a cognitively costly education, both in philosophy and empirical science. This makes the study of scientific representations a natural level of inquiry and an ecological niche for them, after decades of logic-oriented analyses of science. However, if the epistemology of science and computer simulations, in particular, requires delving deep into psychological, social, or computational aspects of scientific processes, an unfortunate combination of different types of expertise is needed to develop it. Furthermore, one cannot expect these epistemological questions about the uses of computers in the empirical science to be disciplinary central for philosophers of mathematics and computer science, sociologists, or psychologists. Naturally, analytically minded epistemologists should hail results showing that aspects or dimensions of computer-based belief-generating processes can be ignored or treated independently. Also, searching where it is easier can be methodologically sound and rational up to a certain point. Nevertheless, epistemologists should guard against the streetlight effect and unjustified simplifications for fear of producing an incomplete and distorted picture of the epistemology of computer simulations.

Attempts to refute extreme or early versions of claims do not provide solid evidence for considering that their moderate versions are totally false. Using such refutations to discard incipient and burgeoning analyses about a novel issue looks like falling prey to confirmation bias. Frigg and Reiss, after rejecting the idea that aspects of the epistemology of simulations are novel, defend a conservative normative stance about which scientific orientations should be adopted. They recommend considering analyses about simulations as merely feeding existing debates, in particular, those about scientific models. Although synergies are needed and overlaps are worth investigating, such a perspective is unduly narrow. Its blind adoption as a communal view may have a chilling effect and distract from important questions that deserve attention, at least if one wants to understand how scientific knowledge is developing at the current time.

Acknowledgements Past interactions with Roman Frigg, Stephan Hartmann, and Paul Humphreys about various issues discussed in this chapter were extremely stimulating, and I probably owe them more than I am aware of. I am also very grateful to the editors, whose comments contributed significantly to improving this chapter.

References

Andersen, H. (2014). *Epistemic dependence in contemporary science: Practices and malpractices*. In L. Soler, S. Zwart, M. Lynch, & V. Israel-Jost (Eds.), *Commentary on epistemic dependence in contemporary science: Practices and malpractices by Hanne Andersen* (pp. 161–173). Routledge Studies in the Philosophy of Science, London: Routledge.

Baker, M. (2016). 1,500 Scientists lift the lid on reproducibility. *Nature News, 533*(7604), 452.

Barberousse, A., Franceschelli, S., & Imbert C. (2009). Computer simulations as experiments. *Synthese, 169*(3), 557–574.

Barberousse, A., & Imbert, C. (2013). New mathematics for old physics: The case of lattice fluids. *Studies in History and Philosophy of Science Part B: Studies in History and Philosophy of Modern Physics, 44*(3), 231–241.

Barberousse, A., & Imbert, C. (2014). Recurring models and sensitivity to computational constraints. Sherwood J. B. Sugden (Ed.), *Monist, 97*(3), 259–279.

Beisbart, C. (2018). Are computer simulations experiments? And if not, how are they related to each other? *European Journal for Philosophy of Science*, 1–34.

Bloor, D. (1976). *Knowledge and social imagery* (Routledge Direct Editions). London, Boston: Routledge & K. Paul.

Collberg, C., & Proebsting, T. A. (2016). Repeatability in computer systems research. *Communications of the ACM, 59*(3), 62–69.

Collins, H. M. (1985). *Changing order: Replication and induction in scientific practice*. London, Beverly Hills: Sage Publications.

De Matteis, A., Pagnutti, S. (1988). Parallelization of random number generators and long-range correlations. *Numerische Mathematik, 53*(5), 595–608.

DeMillo, R. A., Lipton, R. J., & Sayward, F. G. (1978). Hints on test data selection: Help for the practicing programmer. *Computer, 11*(4), 34–41.

DeMillo, R. A., Lipton, R. J., & Perlis, A. J. (1979). Social processes and proofs of theorems and programs. *Communications of the ACM, 22*(5), 271–280.

Demmel, J., & Nguyen, H. D. (2013). Numerical reproducibility and accuracy at exascale. In *2013 IEEE 21st Symposium on Computer Arithmetic* (pp. 235–237).

Dijkstra, E. W. (1978). On a political pamphlet from the middle ages. *ACM SIGSOFT software engineering notes, 3*(2), 14–16.

Dijkstra, E. W. (1972). The humble programmer. *Communications of the ACM, 15*(10), 859–866.

El Skaf, R., & Imbert, C. (2013). Unfolding in the empirical sciences: Experiments, thought experiments and computer simulations. *Synthese, 190*(16), 3451–3474.

Fetzer, J. H. (1988). Program verification: The very idea. *Communications of the ACM, 31*(9), 1048–1063.

Fillion, N., & Corless, R. M. (2014). On the epistemological analysis of modeling and computational error in the mathematical sciences. *Synthese, 191*(7), 1451–1467.

Fomel, S., & Claerbout, J. F. (2009). Guest editors' introduction: Reproducible research. *Computing in Science Engineering, 11*(1), 5–7.

Fresco, N., & Primiero, G. (2013). Miscomputation. *Philosophy & Technology, 26*(3), 253–272.

Frigg, R., & Reiss, J. (2009). The Philosophy of simulation: Hot new issues or same old stew? *Synthese, 169*(3), 593–613.

Frigg, R., & Hartmann, S. (2017). Models in Science. In E. N. Zalta (Ed.), *The Stanford Encyclopedia of Philosophy*, Spring 2017. Metaphysics Research Lab, Stanford University, https://plato.stanford.edu/archives/spr2017/entries/models-science/.

Goldman, A. I. (1999). *Knowledge in a social world*. Oxford, New York: Clarendon Press, Oxford University Press.

Hardwig, J. (1985). Epistemic dependence. *Journal of Philosophy, 82*(7), 335–349.

Hastie, R., Penrod, S., & Pennington, N. (1983). *Inside the jury*. Cambridge, Massachusetts, United States: Harvard University Press.

Heinrich, J. (2004). Detecting a bad random number generator. CDF/MEMO/STATISTICS/PUBLIC/6850. University of Pennsylvania. https://www-cdf.fnal.gov/physics/statistics/notes/cdf6850_badrand.pdf.

Hellekalek, P. (1998). Don't trust parallel Monte Carlo. In *Proceedings Parallel and Distributed Simulation Conference* (pp. 82–89), Alberta, Canada.

Hill, D. R. C. (2015). Parallel random numbers, simulation, and reproducible research. *Computing in Science Engineering, 17*(4), 66–71.

Humphreys, P. (2004). *Extending ourselves: Computational science, empiricism, and scientific method*. Oxford University Press.

Humphreys, P. (2009). The philosophical novelty of computer simulation methods. *Synthese*, *169*(3), 615–626.

Imbert, C. (2014). The identification and prevention of bad practices and malpractices in science. In L. Soler, S. Zwart, M. Lynch, & V. Israel-Jost (Eds.), *Science after the practice turn in the philosophy, history, and social studies of science* (pp. 174–187). Routledge Studies in the Philosophy of Science, London: Routledge

Imbert, C. (2017). Computer simulations and computational models in science. In *Springer handbook of model-based science* (pp. 735–781). Springer Handbooks, Cham: Springer.

Jones, D. (2010). *Good practice in (pseudo) random number generation for bioinformatics applications*. Technical report, UCL Bioinformatics Group.

Kalven Jr, H., & Zeisel, H. (1966). *The American jury*. London: The University of Chicago press.

Kitcher, P. (1992). The Naturalists Return. *The Philosophical Review, 101*(1), 53–114.

Kitcher, P. (1993). *The Advancement of science: Science without legend, objectivity without illusions*. New York: Oxford University Press, 1993.

Kitcher, P. (2002). The third way: Reflections on helen longino's the fate of knowledge. *Philosophy of science, 69*(4), 549–559.

Lenhard, J., forthcoming. Holism, or the erosion of modularity-a methodological challenge for validation. *Philosophy of Science.*

Lenhard, J., & Carrier, M. (2017). *Mathematics as a tool-tracing new roles of mathematics in the sciences*.

Matsumoto, M., Wada, I., Kuramoto, A., & Ashihara, H. (2007). Common defects in initialization of pseudorandom number generators. *ACM Transactions on Modeling and Computer Simulation, 17*(4).

Rennie, D., Yank, V., & Emanuel, L. (1997, August 20). When authorship fails. A proposal to make contributors accountable. *JAMA, 278*(7), 579–585.

Rennie, D., Flanagin, A., & Yank, V. (2001). The contributions of authors. *JAMA, 284*(1), 89–91.

Shapiro, S. (1997). Splitting the difference: The historical necessity of synthesis in software engineering. *IEEE Annals of the History of Computing, 19*(1), 20–54.

Simon, H. A. (1957). *Models of man: Social and rational mathematical essays on rational human behavior in a social setting*. New York: Wiley.

Solomon, M. (1994). Social Empiricism. *Noûs*, *28*(3), 325–343.

Foote, B., & Yoder, J. (1999). *Pattern languages of program design 4 (= Software Patterns. 4)*. Addison Wesley.

Wilson, G., Aruliah D. A., Brown C. T., Hong N. P. C., Davis, M, et al. (2014). Best practices for scientific computing. *PLOS Biology, 12*(1).

Wimsatt, W. C. (2007). *Re-engineering philosophy for limited beings: Piecewise approximations to reality*. Cambridge, Mass: Harvard University Press.

Winsberg, E. B. (2010). *Science in the age of computer simulation*. Chicago: Etats-Unis.

Woods, J. (2013). *Errors of reasoning: Naturalizing the logic of inference*. College Publications.