COMPUTER-BASED METHODS OF KNOWLEDGE GENERATION IN SCIENCE

WHAT CAN THE COMPUTER TELL US ABOUT THE WORLD?

Zur Erlangung des akademischen Grades eines / einer DOKTORS / DOKTORIN DER PHILOSOPHIE (Dr. phil.)

von der KIT-Fakultät für Geistes- und Sozialwissenschaften
des Karlsruher Instituts für Technologie (KIT)
angenommene

DISSERTATION

von

Paul David Grünke

KIT-Dekan / KIT-Dekanin: Prof. Dr. Michael Schefczyk

1. Gutachter/Gutachterin: Prof. Dr. Dr. Rafaela Hillerbrand
2. Gutachter/Gutachterin: Prof. Dr. Richard Dawid

Tag der mündlichen Prüfung: 11.10.2022

Zusammenfassung

Der Computer hat die wissenschaftliche Praxis in fast allen Disziplinen signifikant verändert. Neben traditionellen Quellen für neue Erkenntnisse wie beispielsweise Beobachtungen, deduktiven Argumenten oder Experimenten, werden nun regelmäßig auch computerbasierte Methoden wie ‚Computersimulationen' und ‚Machine Learning' als solche Quellen genannt. Dieser Wandel in der Wissenschaft bringt wissenschaftsphilosophische Fragen in Bezug auf diese neuen Methoden mit sich. Eine der naheliegendsten Fragen ist dabei, ob diese neuen Methoden dafür geeignet sind, als Quellen für neue Erkenntnisse zu dienen. Dieser Frage wird in der vorliegenden Arbeit nachgegangen, wobei ein besonderer Fokus auf einem der zentralen Probleme der computerbasierten Methoden liegt: der Opazität. Computerbasierte Methoden werden als opak bezeichnet, wenn der kausale Zusammenhang zwischen Input und Ergebnis nicht nachvollziehbar ist. Zentrale Fragen dieser Arbeit sind, ob Computersimulationen und Machine Learning Algorithmen opak sind, ob die Opazität bei beiden Methoden von der gleichen Natur ist und ob die Opazität verhindert, mit computerbasierten Methoden neue Erkenntnisse zu erlangen. Diese Fragen werden nah an der naturwissenschaftlichen Praxis untersucht; insbesondere die Teilchenphysik und das ATLAS-Experiment am CERN dienen als wichtige Fallbeispiele.

Die Arbeit basiert auf fünf Artikeln. In den ersten beiden Artikeln werden Computersimulationen mit zwei anderen Methoden – Experimenten und Argumenten – verglichen, um sie methodologisch einordnen zu können und herauszuarbeiten, welche Herausforderungen beim Erkenntnisgewinn Computersimulationen von den anderen Methoden unterscheiden. Im ersten Artikel werden Computersimulationen und Experimente verglichen. Aufgrund der Vielfalt an Computersimulationen ist es jedoch nicht sinnvoll, einen pauschalen Vergleich mit Experimenten durchzuführen. Es werden verschiedene epistemische Aspekte herausgearbeitet, auf deren Basis der Vergleich je nach Anwendungskontext durchgeführt werden sollte. Im zweiten Artikel wird eine von Claus Beisbart formulierte Position diskutiert, die Computersimulationen als Argumente versteht. Dieser ‚Argument View' beschreibt die Funktionsweise von Computersimulationen sehr gut und ermöglicht es damit, Fragen zur Opazität und zum induktiven Charakter von Computersimulationen zu beantworten. Wie mit Computersimulationen neues Wissen erlangt werden kann, kann der Argument View alleine jedoch nicht ausreichend beantworten. Der dritte Artikel beschäftigt sich mit der Rolle von Modellen in der theoretischen Ökologie. Modelle sind zentraler Bestandteil von Computersimulationen und Machine Learning Algorithmen. Die Fragen über die Beziehung von Phänomenen und Modellen, die hier anhand von Beispielen aus der Ökologie betrachtet werden, sind daher für die epistemischen Fragen dieser Arbeit von zentraler Bedeutung. Der vierte Artikel bildet das Bindeglied zwischen den Themen Computersimulation und Machine Learning. In diesem Artikel werden verschiedene Arten von Opazität definiert und Computersimulationen und Machine Learning Algorithmen anhand von Beispielen aus der Teilchenphysik daraufhin untersucht, welche Arten von Opazität jeweils vorhanden sind. Es wird argumentiert, dass Opazität für den Erkenntnisgewinn mithilfe von Computersimulationen kein prinzipielles Problem darstellt, Model-Opazität jedoch für Machine Learning Algorithmen eine Quelle von fundamentaler Opazität sein könnte. Im fünften Artikel wird dieselbe Terminologie auf den Bereich von Schachcomputern angewandt. Der Vergleich zwischen einem traditionellen Schachcomputer und einem Schachcomputer, der auf einem neuronalen Netz basiert ermöglicht die Illustration der Konsequenzen der unterschiedlichen Opazitäten.

Insgesamt ermöglicht die Arbeit eine methodische Einordnung von Computersimulationen und zeigt, dass sich weder mit einem Bezug auf Experimente noch auf Argumente alleine klären lässt, wie Computersimulationen zu neuen Erkenntnissen führen. Eine klare Definition der jeweils vorhanden Opazitäten ermöglicht eine Abgrenzung von den eng verwandten Machine Learning Algorithmen.

Abstract

Computers have significantly changed scientific practice in almost all disciplines. In addition to traditional sources of new knowledge such as observation, deductive arguments, or experiments, computer-based methods such as 'computer simulations' and 'machine learning' are now regularly cited as sources of knowledge. This development in scientific practice raises several questions about these new methods within the philosophy of science, including whether these new methods are in fact suited to serving as sources of new knowledge. This issue is explored in this thesis with a focus on one of the central problems of computer-based methods: opacity. Computer-based methods are considered to be opaque when the causal relationship between input and outcome is not traceable. The thesis addresses three central questions related to the problem of opacity: whether computer simulations and machine learning algorithms are opaque, whether opacity is of the same nature in both methods, and whether opacity prevents computer-based methods from yielding new insights. These questions are explored in close connection with concrete scientific practice; in particular, particle physics and the ATLAS experiment at CERN serve as important examples.

The thesis is based on five articles. In the first two articles, I compare computer simulations with two other methods – experiments and arguments – in order to position them on the 'methodological map' and to identify the challenges of knowledge acquisition that distinguish computer simulations from the other methods. The first article compares computer simulations and experiments. Because of the diversity of computer simulations, however, it becomes clear that a blanket comparison of these two methods is not useful. The article therefore formulates several epistemic aspects that should be used as the framework for specific comparisons in various application contexts. The second article discusses a position formulated by Claus Beisbart that regards computer simulations as arguments. This "Argument View" aptly describes the functioning of computer simulations and thus makes it possible to answer questions about the opacity and inductive character of computer simulations. However, the Argument View alone cannot adequately answer the question of how new knowledge can be obtained with computer simulations. The third article focuses on the role of models in theoretical ecology. Models are central to computer simulations and machine learning algorithms. Questions about the relationship between phenomena and models, considered here using examples from the field of ecology, are therefore central to the epistemic questions of this thesis. The fourth article connects the topics of computer simulation and machine learning. In this article, different types of opacity are defined, and computer simulations and machine learning algorithms are examined based on examples from particle physics in order to determine which types of opacity are present in each case. It is argued that opacity does not pose an intrinsic problem for acquiring knowledge using computer simulations, but model-opacity could be a source of fundamental opacity for machine learning algorithms. In the fifth article, this terminology is applied to chess computers. A comparison between a traditional chess computer and a chess computer based on a neural network allows us to illustrate the consequences of the various types of opacity.

In sum, the thesis offers a methodological classification of computer simulations and shows that an exclusive reference to experiments or to arguments cannot clarify how computer simulations lead to new insights. A clear definition of the opacities present in each case enables differentiation from the closely related machine learning algorithms.

List of individual articles included in the dissertation

Article 1:	Grünke, P. *Computer simulations and experiments: Epistemologically on a par or not?* (submitted)
Article 2:	Grünke, P. & Schweer, J. "How do we learn from Computer Simulations? A Critical Examination of the "Argument View" on Simulation." (submitted)
Article 3:	Grünke, P. & Brinck, K. "Model building in theoretical ecology: Is 'species' a good observable? Or: Why grass is not less important than rabbits." (submitted)
Article 4:	Boge, F. & Grünke, P. "Computer simulations, machine learning and the Laplacean demon: Opacity in the case of high energy physics." forthcoming in Resch, Kaminski, and Gehring (Eds.), *The Science and Art of Simulation II,* Springer (forthcoming) Preprint available: http://philsci-archive.pitt.edu/17637/
Article 5:	Grünke, P. (2019). "Chess, Artificial Intelligence, and Epistemic Opacity." *Információs Társadalom*, *19*(4).

List of figures

Acknowledgements

First of all, I want to thank my supervisor, Rafaela Hillerbrand, for taking me on as a PhD student and offering me the opportunity to do research on the topics I am most passionate about.

The two research groups I was a member of have provided me with my academic home: the PHILETAS group at ITAS and the “The Epistemology of the Large Hadron Collider” group. I am very grateful for the academic discussions as well as for the friends I could make here. A special thanks also goes to the members of the philosophy department at the Stockholm University and Richard Dawid for welcoming and integrating me during my stay in Stockholm. Thanks to The Karlsruhe House of Young Scientist (KHYS) for supporting this research stay.

Jasmin and Max, thank you for all the support from our 312-office group. Florian, Julie and Katharina, thank you for co-authoring papers with me. It has been a joy to discuss and write with you.

Last but certainly not least, my thanks goes to my close friends and my family who have supported me all my life and especially during the rather tough times of finishing this PhD. I could not have done it without you!

Table of Contents

Chapter 1: Introduction

Computer-based methods such as computer simulations or machine learning are ubiquitous in today's scientific practice. Almost every scientific discipline employs these methods and cites them as sources of new knowledge. Their results influence policy decisions such as the closure of European air space after the eruption of a volcano (Webster et al. 2012), the evaluation of various climate change scenarios (IPCC 2014), or the response to the outbreak of a pandemic (Adam 2020). This development in scientific practice raises several questions for philosophers of science. One central question related to computer-based methods is how they fit into the traditional distinction between inductive and deductive methods. Peter Galison describes them as being "at once nowhere and everywhere on the usual methodological map" (Galison 1996, 120). Because models constitute an essential component of computer-based methods, they are very closely connected to theory, while at the same time the practice of simulation is very similar to the practice of experimentation, and new knowledge is often produced via inductive reasoning. Another important question is whether these new methods are suited to serving as sources of new knowledge. It is of central importance for philosophers of science to determine how computer-based methods can generate new knowledge and to explicate some of the major challenges that might hinder such knowledge creation. The scope of these questions is too extensive for conclusive answers to be attained in this thesis; they will certainly require further debate. My aim here is to enrich and advance these debates with a particular focus on the role of opacity in computer-based methods of knowledge acquisition.

It seems natural that computer simulations and machine learning[1] methods should be discussed together, since they share many methodological aspects. As I describe below, machine learning methods could even be understood as a specific type of computer simulation. Both are based on deterministic algorithms that govern every step in their calculations. They are employed to process and analyse large amounts of data and the purposes for which they are used can sometimes be very similar, such as the discovery and prediction of statistical patterns. Of course, the most obvious feature that these computer-

[1] The term machine learning encompasses several different types of algorithms that are used to solve problems by learning from data. For the purposes of this thesis, I will primarily discuss supervised learning algorithms, which will be defined below.

based methods share, in contrast to other tools of scientific inquiry, is the involvement of the computer. This is also the source of one of the most prominently expressed epistemological concerns about computer-based methods: opacity. Opacity is commonly understood as the inability to trace the causal connection between the input and output of a given process.[2] My goal in this thesis is to define precisely what opacity is, what sources of opacity are present in computer simulations and machine learning algorithms, and whether opacity is a fundamental problem for generating knowledge with computer-based methods.[3]

In the remainder of the introduction, I will first present some historical and current examples of the use of computer simulations in science (Section 1), before clarifying the terminology used in this thesis and providing a characterisation of the central elements of the terminology (Section 2). I will then formulate my research questions (Section 3) and give an overview of the answers to these questions found in the chapters that follow (Section 4), before concluding with a discussion of the outlook for future research (Section 5).

1 Computer Simulations in Science

> *"As soon as an Analytical Engine exists, it will necessarily guide the future course of the science."*
> (Babbage 1864: 137)

Charles Babbage was among the first to envision the concept of the digital computer – or "analytical engine" – and he predicted quite accurately that this invention would have a significant impact on almost all areas of science. The introduction of computers and especially their increasing calculation speed in recent decades has enabled scientists to simulate and learn about phenomena that were previously inaccessible (Humphreys 2004).

[2] Paul Humphreys has formulated the most prominent definition of opacity: "[A] process is epistemically opaque relative to a cognitive agent X at time t just in case X does not know at t all of the epistemically relevant elements of the process" (Humphreys 2009: 618). It is interesting to note that Humphreys attaches opacity to processes. Humphreys formulates this definition in the context of computer simulations and parts of the computer-based methods discussed in this thesis can easily be understood as processes, as I elaborate below.

[3] Alongside the question of opacity, questions of uncertainty and robustness pertain to both computer simulations and machine learning, making their comparison also epistemologically interesting. In this thesis, the focus will be on opacity. For work on the uncertainty of computer simulations, see e.g. Parker 2013; Petersen 2012; Hillerbrand 2010; Biddle and Winsberg 2010. For work on the robustness of computer simulations, see e.g. Wimsatt 2007 and Boge forthcoming.

The use of computers in science became wide-spread after World War II, when scientists involved in the Manhattan Project – the American research project that developed and produced the first nuclear weapons – brought their expertise into different disciplines (Benov 2016). Meteorology and particle physics were among the first fields to use computer simulations as a scientific tool and the scientists working in these disciplines were therefore among the first to face the epistemological questions raised by the use of computer simulations (Borelli & Wellmann 2019, Lynch 2008).

The research on nuclear weapons during and after World War II required theoretical calculations that were – though algorithmically calculable in theory – too complex to solve analytically. At the same time, it was not practical to test the weapon designs experimentally. As a solution, Nicholas Metropolis and Stanislaw Ulam, two of the researchers at the Manhattan Project, developed the Monte Carlo method (Metropolis & Ulam 1949). They introduced it as "a statistical approach to the study of differential equations" (335). Sampling outcomes by using random numbers, this method can be used to simulate stochastic processes as well as to calculate approximations to deterministic problems. The Monte Carlo method works independently of computers, but Metropolis and Ulam noted already in their 1949 paper "that modern computing machines are extremely well suited to perform the procedures described" (339). John von Neumann, another colleague at the Manhattan Project, was also fascinated by the possibilities of the Monte Carlo method. All of the above researchers used the newly developed ENIAC, the first electronic digital computer that could be programmed for their Monte Carlo simulations. As Peter Galison (1996) put it, they "built an artificial world in which "experiments" (their term) could take place" (120).

Meteorology was another early discipline to employ computer simulations on a large scale. Already in the 1950s a computer model was used to simulate the dynamics of the atmosphere. The scientists who used these simulations were interested in weather forecasts and the development of the climate as well as advancing their theoretical understanding of the climate system (Parker 2014: 337). The primary impact of the introduction of the computer in meteorology was that it enabled scientists to acquire approximate solutions for the theoretical equations that they assumed governed the climate system, which would have required too much time to calculate by hand. The computer "helped scientists to see what followed (or failed to follow) from the physical assumptions reflected in different mathematical models of the atmosphere or climate system" (Parker 2014: 341). It seems to follow naturally that computer simulations can be seen in this context as a tool for inductive reasoning, as they enabled scientists to test hypotheses about the model of the climate system.

There were a number of practical problems, however. Some of these problems, especially the sensitivity to small variations in input data, can be seen as paradigmatic for computer simulations. Lenhard (2007) discusses the problem of instability in an early model of the atmosphere over long periods, which was later solved by using a mathematical trick that did not conform to physical theory but resolved the stability issue. Another challenge faced by weather forecast models was first made public by Edward Lorenz. Lorenz wanted to repeat a weather simulation, but instead of using the same data as in the previous simulation, he entered the data from a print-out in which the data was rounded after three digits rather than six digits. To Lorenz's surprise, although the variation of the starting values was minor, the simulation result was completely different from the previous simulation run. Lorenz had discovered that nonlinear systems, such as the models of the weather, can behave chaotically. Small changes in the input data increase exponentially during the run of the simulation (Oestreicher 2007).

This is relevant to weather simulations for two reasons. Firstly, due to the discrete representation of numbers in the computer, numbers must be rounded; because of the chaotic nature of the weather system, these rounding errors may lead to significant deviations from the continuous solutions. It also means that in order to reproduce simulation results on a different computer, not just the starting values need to be identical, but also the rounding mechanisms, which is not always easily achievable. Secondly, this discovery of chaotic behaviour imposes a fundamental constraint on the possibility of representing real-world weather conditions using the weather forecast model because of the imprecision of measurement. The relevant input values for the weather model, such as temperature or pressure, cannot be measured to an arbitrary degree of precision and must therefore always approximate the exact value. Since even small deviations from the exact value can lead to significantly different developments in the system, one single simulation run does not necessarily produce an accurate weather forecast. Lorenz's proposed solution was to conduct many simulations with slight variations in the values of the starting parameters. By comparing the different developments and recognizing common patterns among the various simulations, the most likely weather developments can be inferred. This practice of "ensemble forecasting" is the common practice in weather forecasting today (Dizikes 2011).

Another prime example of the use of computer simulations in today's scientific practice is found in the area of high-energy physics: the ATLAS experiment at the Large Hadron Collider (LHC) at CERN.[4] This example also introduces another facet of the use of computer simulations in science. While computer simulations are primarily used as predictive tools in the cases discussed above, in the case of the LHC, computer simulations function as part of an experiment. The ATLAS experiment is designed to collide particles at high energies and observe the resulting phenomena. One result of this experiment was the discovery of the Higgs boson in 2012 (ATLAS Collaboration 2012). Two kinds of simulation are essential for the ATLAS experiment: event generators simulate the collision of two protons and the development of the resulting particles until they reach the detector; detector simulations simulate the reactions of the real-world detector to the particles that result from the collisions and the following decays. The computer simulations of the ATLAS experiment are an excellent case study for the problems of opacity and potential ways of overcoming opacity. The ATLAS simulation infrastructure is a complex combination of various models with so many lines of code that no human being will ever be able to read all of them. At the same time, the LHC physicists have managed to achieve a good deal of transparency in their methods via careful modeling, programming, documenting, and testing (see chapters 2 and 5).

The ATLAS experiment has received a good deal of attention from philosophers because, among other things, computer simulations play a central role in this experiment. One philosophical question that has been discussed by several scholars concerns the (in)dispensability of computer simulations in high-energy physics. Margaret Morrison (2015) has argued that the discovery of the Higgs boson would have been impossible without the use of computer simulations. Other philosophers argue that computer simulations are epistemically equivalent to experiments in the case of ATLAS (Massimi & Bhimji 2015). Others

[4] ATLAS (A Toroidal LHC ApparatuS) and CMS (Compact Muon Solenoid) are the two largest particle detector experiments at the LHC (Large Hadron Collider), which is the largest particle collider in the world.

disagree, arguing against an epistemic parity (Mättig 2021) or for the theoretical possibility of discovering the Higgs boson entirely unaided by computers (Schiemann et al. 2021).

2 Terminology and Analysis

2.1 Computer simulation

There are many different types of computer simulation and it is accordingly difficult, if not impossible, to define them in a way that fits all these types while still being useful. Nevertheless, there are a few distinctions between the various meanings associated with the term ‘computer simulation’ that are central to its understanding.

The first is the distinction between a narrow and a broad perspective on what is meant by ‘computer simulation’ (cf. Winsberg 2019). In the narrow sense, the term ‘computer simulation’ refers to a specific program that is implemented and executed by a computer with the aim of approximating the behaviour of a ‘simulation model’, which typically represents some real-world system. The narrow view defines one run of the program on the simulation model as a computer simulation of the system. That is, the program receives as input the values of all the variables that define a system state and then calculates the system’s development through a predefined number of time steps. It starts with the input data as the system state at t_0 and then calculates step-by-step how the state of the system progresses from there to t_1, t_2, and so on, according to equations given in the simulation model. In my thesis, I refer to this narrow understanding of a computer simulation as a ‘computer simulation run’. Authors such as Beisbart (2012, 2019) use this narrow terminology to focus exclusively on the epistemic questions concerning computer simulations and aim to distinguish these issues from epistemic questions related to models and the modeling process. It is my view that the epistemic questions of computer simulations and models are so closely intertwined that they cannot be fully disentangled from one another. I will therefore take the broader perspective in this thesis. However, the choice to adopt either of the two approaches does not directly impact the answers to the epistemic questions addressed in this thesis.

The broader approach defines 'computer simulation' as a scientific method of studying systems. Proponents of the narrow definition often refer to this as 'computer simulations study'. This definition includes all the elements that are necessary to lead to a computer simulation run, which then produces results that can be used for the relevant scientific purpose; these results will typically constitute knowledge generation in one way or another. Figure 1 illustrates these elements and their relationships, which are discussed only briefly here (a similar figure is described in more detail in chapter 2).

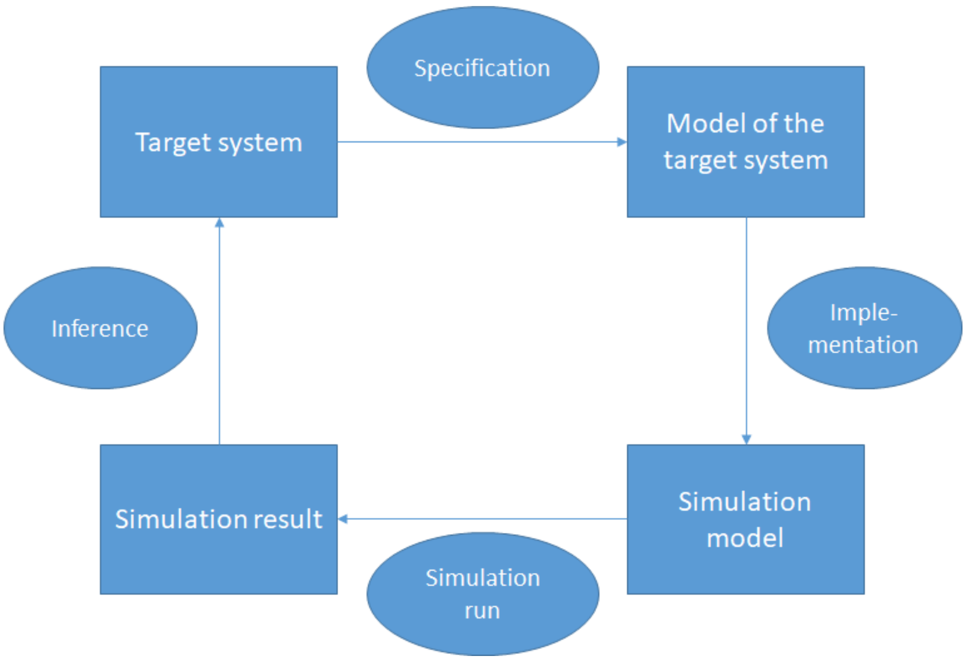

Figure 1: Elements of a computer simulation

The starting point for most computer simulations is some phenomenon that is of interest to the researcher. This phenomenon might be real, such as the collision of particles, or hypothetical, and constitutes the target system. The researcher's goal is to build a simulation model that adequately represents the target system. In order to achieve this, one must first identify which properties of the target system are relevant for the simulation model and on this basis specify a model of the target system. This typically takes the form of mathematical equations. Once this model of the target system is implemented on a computer, the result is the simulation model. Determining whether this simulation model *adequately* represents the target system is one of the central challenges researchers face, and will be addressed in more detail below. The researcher now uses simulation runs (computer simulations in the narrow sense) to create simulation results and thereby gather knowledge about the structure and patterns of the simulation model. In a final step one can then – assuming an adequate representation – apply this knowledge about the model to the target system.

To summarize, in addition to running the simulation, the main elements of this broader understanding of computer simulation are the selection and implementation of an adequate model, the sanctioning of this model's adequacy, and the interpretation of the simulation results, including inferences for the target system.

In both the narrow and the broad interpretation of computer simulation, the use of the computer is central to the definition. An alternative approach taken by Hartmann (1996), among others, is to seek a compositional definition, i.e. to arrive at a definition of 'simulation' first and then define 'computer simulation' as a simulation that is carried out with the use of a computer. Hartmann defines simulation as something that "imitates one process by another process" (83). This approach might be especially useful when focusing on the analogy between the target system and the simulation model. For the purposes of this thesis, however, the role of the computer is central and therefore the definition that specifically targets computer simulation will be more productive.

Another distinction is often made between different types of computer simulations. I will discuss three different types here. Two of the most relevant types are equation-based simulations and agent-based simulations. This distinction refers to the structure of the simulation model. In equation-based simulations, models are typically governed either by discretised differential equations[5] that describe how the different elements of the target system interact, or by a set of equations that describes how the system as a whole evolves over time (Winsberg 2019). Equation-based simulations are typically found in the physical sciences such as particle physics (simulations of particle collisions) or meteorology (weather simulations). In agent-based simulations, rather than having global equations that govern the system's behaviour, models are defined by equations that govern the behaviour of the agents[6] on a local level. This makes them well suited to the study of emergent phenomena.

[5] The computer is a machine that can only calculate discretely. Any continuous equation must therefore be approximated in the modeling phase such that it can be implemented in a discretised form.
[6] Agents represent either individual or collective entities in the target system, such as the players in the simulation of a game, the organisms in an ecosystem simulation or companies in an economics simulation.

Classic examples of agent-based simulations are the Schelling model or the Game of Life (see chapter 3).

A third type of computer simulation consists of stochastic simulations, which are simulations that include variables whose values are governed by individual probabilities. One type of stochastic simulation that has received some attention from philosophers are so-called Monte Carlo simulations (Beisbart & Norton 2012). Monte Carlo simulations do not aim to model a feature of the target system explicitly but rather use a surrogate process that relies on random numbers to approximately calculate the desired feature. One illustrative example of a Monte Carlo simulation is the approximation of π: We take a circle with radius 1 and a square, such that the circle is inscribed in it, with the origin of the coordinate system in the centre of both. Now we use uniformly distributed random numbers to sample N coordinates. Counting the number of coordinates within the square (m) and the number of coordinates within the circle (n) will give us an approximation for the area of the square (which we know to be 4) and for the area of the circle (which we know to be π), and provides an approximation for π: $\pi = 4(n/m)$. The larger the number N of random samplings, the better, on average, the approximation of π will be.

Some philosophers argue that Monte Carlo simulations should not be considered computer simulations because they do not explicitly aim at modeling the target system (Grüne-Yanoff and Weirich 2010). They do, however, fit very well with the above description of computer simulation in Figure 1 and share with the other types of computer simulations the aim of using a substitute system to learn something about the target system.[7] In certain cases, when Monte Carlo simulations are used to simulate systems governed by stochastic dynamical equations, they even fit within the class of equation-based simulations (cf. Beisbart and Norton 2012).

[7] As the Feynman-Kac formula shows, there are also cases, e.g. applications of the heat equation, in which it can be mathematically proven that a stochastic process can act as a representation of the target system (Bertini & Cancrini 1995).

2.2 Machine Learning

Although the terminology used to discuss machine learning methods is usually different, these methods can in some sense be understood as another type of computer simulation. The main difference between the two lies in the way the simulation model is constructed. In the case of the machine learning methods, the simulation model is a neural network.[8] Neural networks are typically represented within the language of graph theory as an arrangement of nodes in different layers, which are connected by directed edges (compare Figure 2). The nodes with no incoming edges comprise the input layer and the nodes with no outgoing edges constitute the output layer, with the nodes between them forming the hidden layers. The nodes of the input layer receive input data, which is then transformed according to functions associated with each of the edges to create new values in the nodes of the next layer. These values will then be transformed according to the functions on the next edges and so on until the nodes of the output layer have values. This process can be understood as analogous to the simulation run, with the values in the output layer analogous to the simulation results.

The aspect of neural networks in machine learning methods that differs critically from the simulation models in the computer simulations described above is the way in which the values are chosen for the functions that transform the input data into output data.

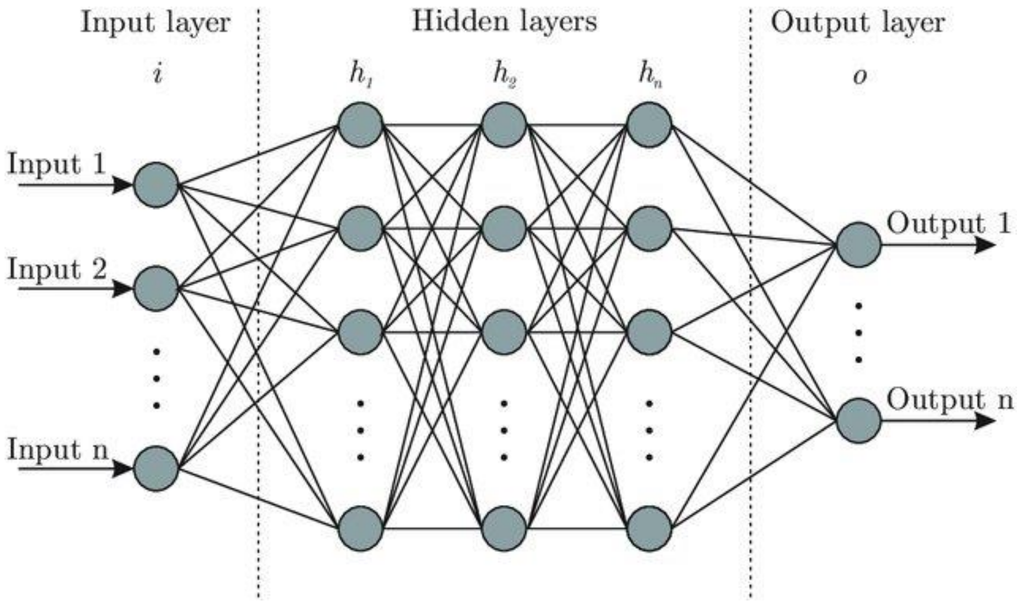

Figure 2: Graph representation of a neural network (Bre et al. 2018)

[8] There are methods that are called machine learning methods that do not use neural networks. These are not discussed in this thesis.

While the simulation models of computer simulations are built by human modelers who choose all the parameter values, the parameters of neural networks in machine learning methods are determined algorithmically in the so-called 'training phase'. During the training phase, the network receives input data for which the desired output data is already known (training data).[9] The goal is for the neural network to produce the desired outputs. To this end, the parameters of the neural network – after being initialised randomly – are iteratively updated during the training phase, optimizing the performance of the neural network with respect to the training data (learning algorithm). One common optimization method is 'stochastic gradient descent', which is described in detail in chapter 5. Once the training phase is completed, the neural network is used to determine the output data for inputs whose output data is unknown.

One could say that in the case of machine learning methods, the simulation model is based on data, while the simulation model in computer simulations is typically based on theory. Of course, the neural network is still a model comprised of equations. These equations, however, are determined during the training phase by the training data and the learning algorithm. One central philosophical question regarding machine learning methods is how the simulation model can be considered a representation of the target system when the equations of the simulation model do not refer directly to anything in the target system. One way to interpret the relation of representation in the case of machine learning methods is via the training data. The neural network is a determined result based on the learning algorithm and the training data. In order for the neural network to successfully produce the correct outputs for new inputs after the training phase, the training data must provide a good representation of the target system. During the training phase, this data set is transformed into a set of equations in the form of a neural network. This is clearly a different kind of representation than is found in the types of computer simulations described above; nevertheless, there are structural similarities between the two. In order to reveal these similarities, let us consider the elements

[9] I describe supervised learning here, which is one popular machine learning method. Other machine learning methods are discussed in detail in chapters 5 and 6.

of Figure 1 once more and try to apply them to the machine learning methods as described here.

The target system is some phenomenon of interest. Identifying the target system, however, is often not as obvious in machine learning methods as it is in the case of computer simulations, because after something is learned about a phenomenon, this knowledge is then applied to some other task. One of the typical applications of machine learning methods is image classification, i.e. identifying what kind of object is present in a specific image. Let us take an example in which a machine learning algorithm is used to determine whether a cat or a dog are present in a specific image. In these cases, one could say that the target system is the visual appearance of a dog or a cat, since this is what the neural network must learn to recognize in order to be able to successfully classify images. The specification process is two-fold. On the one hand, the architecture of the neural network must be specified, which involves among other things decisions about the number of layers, the number of nodes in each layer, and the level of connectivity between the nodes (fully connected, partly connected, only connected forward or also recursively). On the other hand, the training data set must also be specified. This is a very important task because the training data set largely determines how well the resulting neural network will represent the target system. In the case of image classification, one must ensure that the training data covers all kinds of cats and dogs from many different angles to enable the neural network to learn a good representation of the visual appearance of cats and dogs. The more complex the target system is, the more difficult it is to specify a good training data set. Once all of this is specified and a learning algorithm has been chosen, the learning phase can start. This phase is analogous to the implementation of computer simulations; the values of the parameters in the neural network are determined and the neural network, which has the role of the simulation model, is created. The neural network's transformation of input data into output data is equivalent to the simulation run creating simulation results. The type of inference that can be drawn from these results is often different from those of computer simulation. Instead of learning something about the model behaviour in order to learn something about the target system, machine learning methods are often used to produce predictions rather than explanations[10].

2.3 Opacity

One of the philosophical issues discussed prominently in connection with both computer simulations and machine learning methods is the topic of opacity. Simply put, computer simulations or machine learning methods are referred to as opaque or black boxes if we – meaning some epistemic agent – cannot understand the process that leads from the input to the output. Paul Humphreys' (2009) definition of opacity, which has become the standard point of reference for philosophical discussions about opacity, captures this well: "[A] process is epistemically opaque relative to a cognitive agent X at time t just in case X does not know at t all of the epistemically relevant elements of the process." (618)

The involvement of the computer in computer simulations and machine learning methods is one potential source of opacity. Due to the calculation speed of today's computers and the sheer number of lines of code required for a complex computer simulation or a machine learning method, human agents cannot fully comprehend how a certain input leads to a certain output (Saam 2017, Schembera 2017, Kaminski 2018). Results of computer simulations also depend on the many modeling choices that were made during the specification and implementation phases of the simulation model and are further complicated by the dependencies that exist between the different parts of the simulation model. This complexity is an additional source of opacity in computer simulations. The same sources are also present in machine learning methods, with the further complication that the modeling choices for the neural network are not explicitly made by human modelers but rather determined by a complex combination of the training data and the learning algorithm. In this thesis, especially in chapters 5 and 6, I aim to clarify the nature of these opacities and discuss the extent to which they might hinder our ability to generate knowledge about target systems using computer simulations or machine learning methods.

[10] For a wide range of discussion on this topic, see the forthcoming Special Issue on "Machine Learning: Prediction Without Explanation?" in *Minds & Machines*.

3 Research questions

The emergence of computer simulations and machine learning methods raises a broad range of philosophical questions. In this thesis, I aim to address several questions that can be categorized in three different groups.

1) Computer simulations as a source of knowledge: How do computer simulations compare to other scientific methods of knowledge generation, namely experiments and arguments? What epistemic challenges and opportunities do computer simulations share with these methods, and which are unique to computer simulations? Is computer simulation a genuinely new method and does it require its own epistemology?

2) The impact of modeling choices on the potential for knowledge acquisition: What are the differences between theory-based models and data-based models? What are typical problems encountered during the modeling phase? What are good criteria for choosing model parameters?

3) Opacity in computer simulations and machine learning: What kinds of opacities are present in computer simulations and machine learning? Can opacity be overcome in computer simulations and in machine learning? Can we learn anything about target systems from opaque computer simulations or machine learning methods?

4 Overview of the thesis

Chapters 2 and 3 address the questions related to the epistemic status of computer simulations. Chapter 2 discusses whether computer simulations are epistemically on a par with experiments. After reviewing the arguments offered by proponents of various positions, I[11] conclude that the term 'epistemically on a par' is insufficiently defined and, in view of the diversity of experiments and computer simulations, the focus on an epistemic hierarchy between the two methods at such a general level is misguided. There are epistemic questions that require the use of experiments; but there are also epistemic questions for which

[11] Some of the papers that comprise chapters 2 through 6 of the thesis have been co-authored. For the sake of consistency, I will use "I" rather than "we" for all papers. This is in no way meant to diminish the contributions of my co-authors to these papers.

computer simulations are better suited. A generalised determination of the epistemic hierarchy between the two methods is therefore unachievable. Instead, I suggest focusing on the epistemic differences between certain computer simulations and certain experiments in order to evaluate their relative epistemic status. Chapter 3 investigates whether computer simulations can be reconstructed as arguments and whether such a reconstruction fully captures the epistemic power of a computer simulation, as suggested in the “Argument View” proposed by Claus Beisbart. Computer simulations share many features with formal arguments and it is indeed possible to reconstruct a simulation run as an argument. This reconstruction does not, however, provide an entirely satisfying account of how new knowledge is generated through computer simulations, since new knowledge is often produced not by one single simulation run, but rather derives from the inductive reasoning that is possible only after acquiring the results of many simulation runs. Reconstructing the simulation run as an argument does, however, offer insights into questions related to the opacity of computer simulations. Chapters 2 and 3 show that while computer simulations exhibit some features that are characteristic of experiments and some that are characteristic of formal arguments, they also differ from both of these methods.

Chapter 4 considers an example from the field of theoretical ecology in order to examine the role of models and their relationship with data and observation. As models are central to both computer simulations and machine learning algorithms, questions concerning the relationship between phenomena and models are highly relevant to the epistemic questions of this thesis. In contrast to the models employed in climate science and particle physics, the development of models in theoretical ecology are typically based on observational data rather than theories. This poses new challenges for our inquiry. In particular, the potential biases in observational data – discussed in chapter 4 in connection with the observable ‘species’ in ecological systems – are relevant to machine learning methods, which are crucially dependent on high quality data sets for both the training phase and the test phase.

Chapters 5 and 6 address questions concerning the opacity of computer simulations and machine learning methods. Chapter 5 discusses opacity in the context of high-energy physics. Here, I distinguish between various types of opacities, establish the difference between contingent and fundamental opacity, and show that there is no fundamental opacity in computer simulations. This means that there is no intrinsic reason that opacity should hinder an epistemic agent from understanding how the input into a computer simulation is transformed into the output. There might only be restrictions such as limitations of time or of access to information that prohibit an epistemic agent from resolving all the opacity. This is also the case for most sources of opacity in machine learning methods. I do, however, argue that model-opacity – the inability to arrive at a unique theoretical model underlying a successful machine learning method – is a candidate for a fundamental opacity. Chapter 6 makes a similar argument in another domain, namely chess engines. I compare Stockfish, a chess engine which is modelled in the same way as a computer simulation, with AlphaZero, a chess engine based on machine learning. AlphaZero is the more successful chess engine – that is, it is better at playing chess. However, while the opacity of Stockfish can be fully resolved, AlphaZero remains opaque, making it more difficult to understand why the latter is more successful than the former.

5 Outlook

The introduction of the computer into scientific practice has undoubtedly changed science itself significantly. Nearly all scientific disciplines use computer simulations today and some disciplines only exist because of the possibilities created by the computer.

The epistemological questions surrounding computer simulations have not yet been fully resolved. It is my hope that this thesis contributes to a more complete epistemology of computer simulations, thus enabling scientists to have greater confidence in using computer simulations as a scientific method.

As machine learning methods have become popular fairly recently and philosophers began to engage with this topic only a short time ago, there are several open epistemic questions in this area, especially concerning opacity and the relationship between the neural network, the target system, and the possibility of understanding. Machine learning algorithms, more so than computer simulations, also raise ethical questions. They may be applied in many contexts in which decisions traditionally require explanations that machine learning algorithms cannot yet provide. It is essential that the epistemic questions of machine learning be resolved in order to allow for a well-informed evaluation of the related ethical questions.

The relatively new field of Explainable AI might offer a good approach to bridging the epistemic and ethical questions of machine learning.

References

Adam, David. 2020. “Special report: The simulations driving the world's response to COVID-19.” *Nature* 580.7802: 316-319.

ATLAS Collaboration. 2012. “Observation of a new particle in the search for the Standard Model Higgs boson with the ATLAS detector at the LHC.” *Physics Letters B* 716, no. 1: 1–29. doi:10.1016/j.physletb.2012.08.020.

Babbage, Charles. 1864. *Passages from the Life of a Philosopher*. London: Longman, Green, Longman, Roberts and Green.

Beisbart, Claus. 2012. “How Can Computer Simulations Produce New Knowledge?.” *European Journal for Philosophy of Science* 2(3): 395–434.

Beisbart, Claus. 2019. “What Is Validation of Computer Simulations? Toward a Clarification of the Concept of Validation and of Related Notions.” In: *Computer Simulation Validation*, Claus Beisbart and Nicole J. Saam (eds.), 35–67. Cham: Springer International Publishing.

Beisbart, Claus, and John D. Norton. 2012. “Why Monte Carlo simulations are inferences and not experiments.” *International Studies in the Philosophy of Science* 26.4: 403-422.

Benov, Dobriyan M. 2016. “The Manhattan Project, the first electronic computer and the Monte Carlo method.” *Monte Carlo Methods and Applications* 22, no. 1: 73-79.

Bertini, L., and N. Cancrini. 1995. “The stochastic heat equation: Feynman-Kac formula and intermittence.” *J Stat Phys* 78: 1377–1401.

Biddle, Justin, and Eric Winsberg. 2010. “Value judgements and the estimation of uncertainty in climate modelling.” In: *New Waves in Philosophy of Science*, P. D. Magnus & Jacob Busch (eds.), 172—197. London: Palgrave-Macmillan.

Boge, Florian J. forthcoming. “Why Trust a Simulation? Models, Parameters, and Robustness in Simulation-Infected Experiments.” *British Journal for the Philosophy of Science*.

Borrelli, A., and J. Wellmann. 2019. “Computer Simulations Then and Now: an Introduction and Historical Reassessment.” *N.T.M.* 27: 407–417

Bre, Facundo, Juan M. Gimenez, and Víctor D. Fachinotti. 2018. "Prediction of wind pressure coefficients on building surfaces using artificial neural networks." *Energy and Buildings* 158: 1429-1441.

Dizikes, Peter. 2011. "When the butterfly effect took flight." *MIT Technology Review–digital edition.* Massachusetts 2. https://www.technologyreview.com/2011/02/22/196987/when-the-butterfly-effect-took-flight/

Frigg, Roman, and Julian Reiss. 2009. "The philosophy of simulation: hot new issues or same old stew?." *Synthese* 169.3: 593-613.

Galison, Peter. 1996. “Computer simulations and the trading zone.” In: *The Disunity of Science: Boundaries, Contexts, and Power*, Peter Galison & David J. Stump (eds.), 118—157. Palo Alto, CA: Stanford University Press.

Grüne-Yanoff, Till, and Paul Weirich. 2010. "The philosophy and epistemology of simulation: A review." *Simulation & Gaming* 41.1: 20-50.

Hartmann, Stephan. 1996. “The world as a process.” In: *Modelling and Simulation in the Social Sciences from the Philosophy of Science Point of View*, R. Hegselmann, U. Mueller, and K.G. Troitzsch (eds.), 77-

100. Theory and Decision Library (Series A: Philosophy and Methodology of the Social Sciences), vol 23. Dordrecht: Springer.

Hillerbrand, Rafaela. 2010. "On non-propositional aspects in modelling complex systems." *Analyse & Kritik* 32.1: 107-120.

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

Humphreys, Paul. 2009. "The Philosophical Novelty of Computer Simulation Methods." *Synthese* 169(3): 615–26.

IPCC. 2014. *Climate Change 2014: Synthesis Report. Contribution of Working Groups I, II and III to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change [Core Writing Team, R.K. Pachauri and L.A. Meyer (eds.)]*. Geneva, Switzerland: IPCC.

Kaminski, A. 2018. "Der Erfolg der Modellierung und das Ende der Modelle. Epistemische Opazität in der Computersimulation." In: *Technik – Macht – Raum. Das Topologische Manifest im Kontext interdisziplinärer Studien,* Andreas Brenneis, Oliver Honer, Sina Keesser & Silke Vetter-Schultheiß (eds.), 317-333. Wiesbaden: Springer.

Lenhard, Johannes. 2007. "Computer simulation: The cooperation between experimenting and modeling." *Philosophy of Science* 74.2: 176-194.

Lynch, P. 2008. "The origins of computer weather prediction and climate modeling." *Journal of computational physics* 227(7): 3431-3444.

Mättig, P. 2021. "Trustworthy simulations and their epistemic hierarchy." *Synthese*, 1-32.

Metropolis, Nicholas, and Stanislaw Ulam. 1949. "The Monte Carlo method." *Journal of the American statistical association* 44.247: 335-341.

Morrison, M. 2015. *Reconstructing reality: Models, mathematics, and simulations*. Oxford Studies in Philosophy of Science. Oxford: Oxford University Press.

Oestreicher, Christian. 2007. "A history of chaos theory." *Dialogues in clinical neuroscience* 9.3: 279.

Parker, Wendy S. 2013. "Ensemble modeling, uncertainty and robust predictions." *Wiley Interdisciplinary Reviews: Climate Change* 4.3: 213-223.

Parker, Wendy S. 2014. "Simulation and understanding in the study of weather and climate." *Perspectives on Science* 22.3: 336-356.

Petersen, Arthur C. 2012. *Simulating nature: a philosophical study of computer-simulation uncertainties and their role in climate science and policy advice*. Boca Raton: CRC Press.

Saam, N. J. 2017. "Understanding social science simulations: Distinguishing two categories of simulations." In: *The Science and Art of Simulation I: Exploring – Understanding – Knowing*, M. M. Resch, A. Kaminski, P. and Gehring (eds.), 67–84. Cham: Springer International Publishing.

Schembera, B. 2017. "Myths of simulation." In: *The Science and Art of Simulation I: Exploring – Understanding – Knowing*, M. M. Resch, A. Kaminski, P. and Gehring (eds.), 51–63. Cham: Springer International Publishing.

Schiemann, G., C. Zeitnitz, and M. Krämer. 2021. *Experimental high-energy physics without computer simulations: Limits of theory dependence*. Manuscript in preparation.

Webster, H. N., et al. 2012. "Operational prediction of ash concentrations in the distal volcanic cloud from the 2010 Eyjafjallajökull eruption." *Journal of Geophysical Research: Atmospheres* 117.D20.

Wimsatt, William C., and William Kurtz Wimsatt. 2007. *Re-engineering philosophy for limited beings: Piecewise approximations to reality*. Cambridge, MA: Harvard University Press.

Winsberg, Eric. 2019. "Computer Simulations in Science." *The Stanford Encyclopedia of Philosophy* Edward N. Zalta (ed.). URL = <https://plato.stanford.edu/archives/win2019/entries/simulations-science/>.

Chapter 2: Computer simulations and experiments: Epistemologically on a par or not?

Abstract

A central issue in the debate about the epistemology of computer simulations concerns the question whether a computer simulation can be epistemologically on a par with an experiment. In this paper, I present a reconstruction of the process of modelling, and testing a computer simulation. Based on this I structure and discuss the arguments of the debate about epistemic hierarchy between computer simulations and experiments. Using examples from high-energy physics, I argue that the focus on epistemic hierarchy between computer simulations and experiments in this generalised form does not do justice to the specific epistemic differences. I advocate a more differentiated view on the epistemic advantages and disadvantages of computer simulations and offer epistemic aspects that would allow for a more meaningful distinction.

1 Introduction

Computer simulations have become one of the most important tools in today's science. Researchers as well as decision-makers place more and more trust in the results of computer simulations, be it in climate science, neuroscience, or high-energy physics. Philosophers of science discuss the epistemic status of computer simulations in order to assess whether this trust is justified.

A central issue in the debate about the epistemic status of computer simulations focuses on the comparison between the epistemic status of computer simulations and the epistemic status of experiments (e.g.: Guala 2002, Massimi and Bhimji 2015, Morgan 2005, Morrison 2009, Parke 2014, Parker 2009, Winsberg 2010). Highlighting similarities and differences between computer simulations and experiments helps place computer simulations on the "methodological map" (Galison 1996, p. 120).

The overall goal of the debate seems to be to arrive at a widely agreed epistemology of computer simulations. In this paper, I will structure the current debate along some of the major arguments. A comparison between experiments and computer simulations without concrete contexts is difficult and possibly not very meaningful, since both ‘experiment’ and ‘computer simulation’ are terms that cover a very diverse set of practices. This makes it difficult to find a characterisation that fits all of them and is still informative about their epistemic value. High-energy physics and especially the ATLAS and CMS experiments at the Large Hadron Collider[12] are scientific endeavours in which computer simulations abound and are used for a wide range of purposes, such as the design of detectors, the analysis of data or the estimation of ‘backgrounds’ in experiments.[13] This makes high-energy physics a fruitful study object to learn about the epistemology of computer simulations. I will use examples from the scientific practice in high-energy physics to point to epistemic differences between computer simulations and experiments throughout this paper.

Philosophers of science disagree about the conceptual nature of computer simulations. Some view computer simulations as numerical experiments (Humphreys 2004), as model experiments (Morgan 2005), or as computer experiments (Gramelsberger 2010). Others argue that simulations should be treated as arguments (Beisbart 2012). As Boge (2019) has shown, these accounts do not necessarily disagree and I will not follow any of these accounts explicitly, but rather discuss computer simulations based on my reconstruction of their modelling and testing that is provided in the following section.

In the following I aim to show that a central problem in the debate about computer simulations is that one of the key terms – “epistemologically on a par” (and correspondingly also the terms “epistemically superior/inferior”) – is not defined explicitly in any of the relevant articles and is used with different meanings. Morrison (2009), for example, argues that in specific situations simulation and experiment can be epistemologically on a par and can both provide measurements because there are no epistemically relevant differences between them. She argues for the central role of models in both simulations and experiments and concludes that materiality cannot justify an epistemic privilege of experiments. Massimi and Bhimji (2015) use “on a par” synonymously for “interchangeably used”. Morgan (2005) states, “[w]e are more justified in claiming to learn something about the world from the experiment” (p. 323), arguing for an epistemic privilege of experiments over simulations. None

[12] ATLAS (A Toroidal LHC ApparatuS) and CMS (Compact Muon Solenoid) are the two largest particle detector experiments at the LHC (Large Hadron Collider), which is the largest particle collider in the world.

[13] Consult e.g. ATLAS Collaboration (2010) for an overview over the different purposes of computer simulations in the ATLAS experiment.

of the authors offers a definition for “epistemologically on a par” and they remain vague about the consequences of the epistemic statuses of computer simulations and experiments. Even though all of them use the same term, they seem to talk about different things.

Arguments are made for an epistemic superiority of experiments over computer simulations as well as for computer simulations being epistemologically on a par. My goal in this paper is to show that even though the two argumentations come to different conclusions, they are both sound. This is, however, no contradiction, since they start from different perspectives on what it means for two methods[14] to be epistemologically on a par. I will offer different interpretations of the term “epistemologically on a par” to illustrate these perspectives. One line of arguments focuses only on the methods (methodological perspective); the other one incorporates the epistemic community in which the methods are used (pragmatic perspective). Both perspectives are, as I contend, useful to understand the relationship between experiments and simulations. The term “epistemologically on a par” however should ideally be abandoned and replaced by more descriptive terms that describe the epistemically relevant distinctions between computer simulations and experiments that are subsumed under the argument about an epistemic hierarchy in the present debate.

In the next section I will provide a reconstruction of the process of modelling and testing of a computer simulation. This reconstruction will be used to structure the debate on the epistemic status of computer simulations and experiments. I will assess the arguments for the two positions – an epistemic privilege of experiments (Section 3) and experiments and simulations being epistemologically on a par (Section 4). In Section 5 I will discuss the two perspectives in the context of examples of computer simulations in high-energy physics. From this result, I will identify epistemically relevant differences between experiments and computer simulations and suggest terms that allow for a more differentiated distinction between the epistemic value of computer simulations and experiments (Section 6).

[14] In the context of this paper, I will use the term ‘method’ to refer to computer simulations and experiments in the way I introduced them above. The arguments could however also be made for another set of methods and especially the characterisations in Section 6 can also be read with a broader interpretation of the term ‘method’.

2 Computer simulation – a reconstruction

In order to gain a basis for the understanding of the different arguments for the epistemic status of computer simulations, I reconstruct the way in which computer simulations are modelled and the role of the testing processes that are used to generate trust in the results of computer simulations. Identifying the relevant steps of modelling and testing a computer simulation, I will show that experiments play a crucial role in the validation of the connection between the simulation and the phenomenon that is investigated.[15]

Figure 3 depicts the different entities (rectangular in the figure) and processes (oval in the figure) that are part of this reconstruction to explain the phases of modelling and testing of a computer simulation. In the following I will discuss the different entities and processes and use a toy example – the throwing of a baseball and the calculation of its trajectory – for illustration.

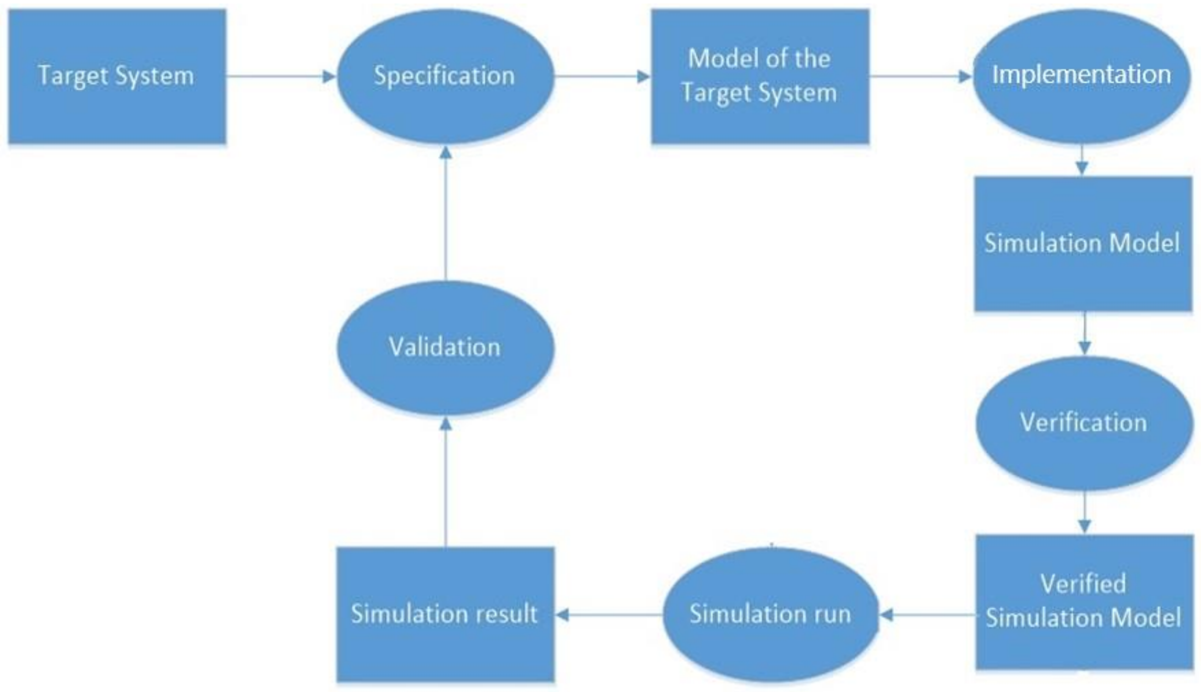

Figure 3: Schematic description of the phases of modelling and testing of a computer simulation. The rectangles in the figure represent entities in the process; the ovals describe process steps by the modeler. The arrows describe the general direction of the process; there are, however, many possible loops in the actual process.

[15] The construction of a computer simulation will in practice of course not always strictly follow the scheme I develop here. I do, however, believe that all the steps will usually be at least implicitly present.

The figure is similar to reconstructions by other authors.[16] The entity *model of the target system* is usually not formulated as an entity of its own but subsumed in either the target system or the simulation model. For a distinction between verification and validation it is, however, helpful to have it separated as I describe below. Verification and validation feature prominently in my reconstruction, because they are the main processes that provide trust in the results of computer simulations and thereby epistemic value. In my understanding of verification and validation I follow Sargent (2013) (many other definitions are very similar): Verification is designed to ensure that the simulation is implemented correctly and works as specified. Validation is designed to test whether the simulation reproduces the data of the simulated phenomenon as accurately as intended.

Let us now discuss the individual elements of Figure 3. The starting point of most simulations is a phenomenon of interest to the researcher. The goal is to build a simulation model that represents the relevant parts of the phenomenon and its context adequately. If this is the case, the researcher can use simulations to learn about the model and then use this knowledge to infer something about the behaviour of the phenomenon from the behaviour of the model. The phenomenon is what I call the *target system* in Figure 3.

The first step in the process of building a simulation model is to specify a *model of the target system*. The term 'target system' is usually used in the debate to stand in for whatever the model shall represent. It is important, however, to distinguish the target system from the model of the target system. Whilst the target system can be almost anything we can describe with natural language, the model of the target system is something more formal. The formalisation is necessary for the implementation. The model of the target system has to be describable by a computer language. The researcher has to specify which entities or processes are used for a representation of the target system. This is not a deterministic choice, but one that will be guided by experience and knowledge about the target system and possibly also by intuition or pragmatic motivations.

In the toy example I will use here, the target system is the phenomenon of throwing a baseball. The researcher wants to predict the trajectory of the baseball. For the model of the target system one has to specify which properties of the ball will be a part of the model (the

[16] Compare e.g. Boge (2019), Sargent (2013), Oberkampf and Barone (2006) or Oberkampf et al. (2004).

modelling choice can range from the ball being modelled as a point-mass to a detailed modelling of shape, spin, and structure of the surface of the ball) and which forces are included in the model (e.g.: gravity, aerodynamic forces). As can be seen from this example already, choosing the target system does not lead to a straightforward definition of the model of the target system. Many different models of the target system are imaginable, which could be adequate for different contexts.

Once the model of the target system is specified, in the next step – the *implementation* – the chosen specifications have to be implemented to create a *simulation model*. In this step, errors and uncertainties can occur due to practical constraints. The computer is a machine that can only calculate discretely, therefore every equation that was written in a continuous way in the model of the target system has to be discretised for the implementation.[17] Another source of errors are constraints due to limited calculating power or calculating time. Abstractions and idealisations have to be introduced in the simulation model to accommodate for that.[18]

There exists a growing literature about how the modelling should work and how verification processes can be designed to test whether the resulting simulation model fulfils the requirements specified by the model of the target system[19] (Oberkampf and Roy 2012, Roy 2005, Sargent 2013). How convincing the arguments are that the simulation model is an adequate representation of the model of the target system is not the question I want to focus on in this paper. Frigg and Nguyen (2017) offer a comprehensive overview on the topic, reviewing different accounts of how models can be understood and working out respective definitions of what a model has to fulfil to be seen as a (adequate) representation. For the purpose of this paper, I will assume that it is possible to verify the simulation model.[20]

[17] Mathematicians have developed various techniques for discretisation – Morrison mentions "finite element methods, finite difference methods, particle methods" (Morrison 2015, 254) as some of the methods.

[18] Compare e.g. Winsberg 2010, p. 7-28 for examples.

[19] The verification can justify the choices that have been made concerning abstractions, idealisation, discretisation etc. during the modelling process for the implementation of the specifications of the model of the target system into a simulation model. Verification, however, cannot help to justify the first step, the specification of the model of the target system.

[20] Oreskes et al. (1994) argue that verification of "numerical models of natural systems is impossible" (641). They claim that "[t]o say that a model is verified is to say that its truth has been demonstrated" (641). They conclude that "[i]n its application to models of natural systems, the term verification is highly misleading", since it "suggests a demonstration of proof that is simply not accessible" (642). Even without going into the details of the question of what it would mean to say that 'the truth of a model has been demonstrated', it seems reasonable to assume that this is not the aim of the technical term 'verification' as can also be seen in (Roy 2005, Oberkampf and Roy 2012, Sargent 2013).

Once there is a verified simulation model, meaning that we can be confident that everything specified in the model of the target system is implemented in the simulation model and works as intended, the next step is the *simulation run*. A number of technical challenges can introduce uncertainties in this step. Apart from the possibility of human errors, there are also more principled problems, mostly originating from the fact that the computer has to calculate discretely and in the internal representation, numbers can only be computed and saved with a limited number of digits. Numbers therefore have to be cut off or rounded and the errors that are created by this can lead to significant deviations from the correct result if large amounts of computations are made.[21] The simulation run produces data – the *simulation results* – which are ideally in the same format and include the same variables as the data that can be gathered from the target system via experiments.

The final step in the reconstruction is the *validation*. The choices we made for the model of the target system in the very beginning (*specification*) cannot be verified within the modelling process. They have to be validated by a comparison between the simulation results and data gathered from measurements and experiments that were made on the target system. In the toy example this means that we have to go outside and throw the baseball, measure the trajectory and compare it to the predictions that the simulation results provide. The validation step is not possible within the method of simulation as was the case for verification. External validation through experiments (or other means of data collection such as measurements or observations) is necessary to validate the choices that have been made prior to the modelling process and thereby confirm the adequateness of the simulation model for the target system.[22]

Based on this reconstruction of the modelling and testing of a computer simulation, I will now structure and discuss the arguments in the debate about the epistemic hierarchy between computer simulations and experiments.

[21] Thomas Ludwig presented this claim at the "SAS Workshop 2018 – Epistemic opacity in computer simulation and machine learning". Cf. also Oestreichers (2007) discussion of Lorenz' discovery of the chaotic behaviour of weather models.

[22] Experimental practices have their own difficulties of course and a good understanding of the measurement apparatus is needed. This is the topic of a wide range of work and will not be included in this paper. Compare e.g. Hacking 1983, Franklin 1986, Galison 1987.

3 Epistemic privilege of experiments – the methodological perspective

In the following, I will structure the debate about the epistemic status of computer simulations by presenting and assessing three of the most relevant arguments for an epistemic privilege of experiments over computer simulations.

I start with the "materiality argument". Morgan (2005) argues in the context of laboratory experiments that experiments have an ontological privilege, which translates into an epistemic privilege:

We are more justified in claiming to learn something about the world from the experiment because the world and experiment share the same stuff. In contrast, inference from the model experiment is much more difficult as the materials are not the same – there is no shared ontology, and so the epistemological power is weaker. (Morgan 2005, p. 323)

"Same stuff" is a very loose formulation.[23] Let us try to understand what Morgan means with it. Following Harré, Morgan has experiments in mind that are of the same materials, yet "simpler, more regular and more manipulable" (Harré 2003, p. 27) in their laboratory version than in their natural environment. One can see how the use of the same physical material can be helpful if there exists little knowledge about the phenomenon as, for example, in the case of the testing of new drugs. It seems clear that testing drugs on animals or, if possible, on humans is epistemologically preferable to computer simulations. The assumption that seems to underlie this argument is that with the "same stuff" involved, the same causal mechanisms are automatically at work in experiment and phenomenon. In the simulation, on the other hand, only the mechanisms, which are implemented in the model will be at work, and these are not necessarily the same as the ones in the phenomenon.

It is however not clear what defines two materials to be same materials. The sameness would have to originate from shared properties, but it is unclear which of the properties should be chosen. Take the example of drug testing. In some instances, an animal might share enough relevant properties with humans to treat them as being of the same material. In other instances, it might not even be reasonable to speak of one human as being from the same material as another human in the context of drug testing. The underdetermination of this

[23] One could argue that an intuitive understanding of "same stuff" might work. However, I believe that basing the epistemic hierarchy on an intuition about the term "same stuff" would lead to misunderstanding due to differing intuitions.

notion of being from the same material is as much of a problem as the one of “shared stuff”. If it boils down to the necessity of defining the relevant properties for each case, nothing is gained by introducing an argument that is based on two things being from the same material. This leads to the conclusion that the materiality argument should not be considered as an epistemic argument, but rather as a heuristic. Using the same materials that constitute the target system might be heuristically advantageous, but since what qualifies as the same material has to be defined, the problem of identifying the epistemically relevant aspects of the target system remains.

A second argument for an epistemic privilege of experiments is the “surprise argument”. This argument is made in different strengths, but always comes down to the statement that simulations cannot surprise in the same way experiments can. One could argue that simulations can surprise the researcher based on his epistemic state, but as Parke (2014) points out, this could lead to the undesirable situation that we would have to classify the same simulation as surprising or not surprising, depending on the researcher’s epistemic state. Therefore, an argument for an epistemic privilege of the experiment based on the ability to surprise should be made independently of the researcher’s background knowledge.

In a simulation the researcher who uses the simulation can be surprised by what the simulation result reveals about the consequences of the model decisions and input parameters. The user might be surprised by the result even if she is the one who wrote the simulation because the process by which the simulation result is reached can be epistemically opaque.[24] This can be the case because the computations by the computer are made significantly faster than calculations could be done by humans and the user of the computer simulation is usually not able to examine every step of the computational process that leads to the result of the simulation. In principle, however, such an examination would be possible and all the simulation results can be explained by the modelling choices. All the epistemically relevant elements are available to the user and an epistemic agent with unlimited resources

[24] I follow Humphreys‘ definition of epistemically opaque: “[A] process is epistemically opaque relative to a cognitive agent X at time t just in case X does not know at t all of the epistemically relevant elements of the process.” (Humphreys 2009, p. 618)

and calculation speed would not be surprised by any computer simulation result (Boge & Grünke forthcoming).[25]

This is different in the case of the experiment. Even though the result of the experiment is also determined by the experimental setup, the mechanisms that are involved are not programmed by the experimenter and cannot be accessed directly. There might exist hidden mechanisms, which become relevant and can be discovered in the process of experimenting. Events or patterns might occur that do not conform to any previous experiences. The history of physics offers many examples of surprising experimental results that led to discoveries and new theories (some examples: the Michelson-Morley experiment, the Rutherford gold foil experiment, or the double-slit experiment). The surprise argument points to an epistemic difference between experiments and computer simulation. There might exist principled opaqueness in the case of experiment, but not in the case of computer simulations. Surprise in the case of computer simulations would be a surprise about the deductive consequences of the initial conditions. Surprise in the case of experiments can be caused from a feature of nature the experimenter did not and could not know about.

A third argument can be based on the role of experiments in the above presented reconstruction (Figure 3) – the argument of epistemic priority. If an experiment produces a result, this result can be validated with the same method by recreating the same experimental setup and testing whether the result still shows up. A reproduction of the results from a second experimental setup increases the trust in the result significantly. The same process would be of nearly no effect for a simulation. Rerunning the same code on the same computational model and getting the same result would not strengthen the trust in the correctness of the simulation result in the same way that it does for an experiment. Rerunning a computer simulation with exactly the same parameters can check whether the result was an artefact of the computer, especially if it is rerun on a different computer.[26] As a further testing mechanism, it is also possible to check whether the result is robust against small

[25] Imagine a Laplacean demon that comprehends all the lines of code and is able to calculate all of the determined consequences instantly.

[26] Ludwig gives an example of a climate simulation that is run four times with exactly the same configuration and produced four similar yet not identical results. In this example it is not completely clear from what the differences in results originate and the general assumption is still, that rerunning the same simulations should lead to getting the same results (Presentation *at the "SAS Workshop 2018 – Epistemic opacity in computer simulation and machine learning").*

changes of the input parameters. None of these results will however further the trust in the real-world occurrence of the results. This can only be gained via an experiment.

This means that in order to argue for drawing an inference from the simulation result to the phenomenon the use of experimental methods is necessary, while in the case of an experiment, no additional methods are necessary. Winsberg (2009) describes this as "epistemological priority of experiments" (p. 591). The knowledge we need to learn something from a simulation is knowledge that we gained from experiments and observation.

To summarize this section: There are three main arguments for those who argue for an epistemic privilege of experiments. The materiality argument is based on the practical assumption that we can identify materials as the same materials and that this sameness translates into the same causal mechanisms being at work in experiment and target system. Since it is not possible to pin down the sameness concretely, I consider this as a heuristic and not an epistemic argument. The surprise argument can be understood as an in-principle argument. Experiments and computer simulations can both surprise their users. In the case of the experiment however, this could be a surprise that not even an epistemic agent with unlimited resources could have predicted confidently. The argument for an epistemological priority of experiments points to the fact that computer simulations are dependent on the results of experiments to generate trust in their results. Following the methodological perspective, it seems therefore very reasonable to speak of an epistemic privilege of experiments.

4 Epistemologically on a par – pragmatic perspective

The same reconstruction (Figure 3) can be considered from what I want to call a 'pragmatic perspective'. This perspective includes the knowledge that exists in the epistemic community, which is gained from previous experiments, and the theories that have evolved from them. In the case of throwing balls of a certain shape there is a lot of experience and data[27], and the choices for the model of the target system are already validated for a number of specific contexts. Therefore, at this point in time, if we want to find out what happens to the trajectory

[27] Complex computer simulation programs exist e.g. for the analysis of curve balls in baseball as well as for many other sports (Miller 2013).

of the baseball if we change the direction in which we throw the baseball by a few degrees we do not have to go outside anymore and throw the ball. Based on the large amount of data we can confidently use a verified and validated simulation model and trust that the simulation results are good descriptions of what would happen in the real world. In this context it does make sense to say that the simulation is epistemologically on a par with experiments.

Morrison (2009), Parker (2009) and others argue against the materiality argument by saying that in both experiments and simulations there is a target of interest and an object, which is investigated and intervened on. Therefore, both in experiments and simulations, it has to be argued why knowledge that is gained about the object that is used as a stand-in for the target of interest can be transferred onto the target of interest itself. They argue against Morgan's position that the same materiality justifies this in the case of the experiment ("Materiality is not enough"). They do not problematize the definition of "same stuff", but claim that the involvement of the "same stuff" does not guarantee that the inferences back to the world are more justified. There might be pragmatic reasons for that, as in the case of weather forecasting where it is impossible to create a laboratory experiment that consists of the "same stuff" and replicates the actual configuration of the atmosphere adequately to create predictions from it (Parker 2009). More principled reasons could be fundamental limitations of measurement accuracy, which limit experiments for example in the case of cosmology or fluid mechanics. Simulations can be more exact in their predictions in these fields in some contexts. In both computer simulations and experiments, an analogy between the target and the stand-in has to be drawn and it is not clear a priori why this should be easier in an experiment than in a simulation. It is rarely contested that computer simulation can be more useful for specific practical purposes. This should not be mixed up with the epistemic value of the results however.

Morrison (2009) argues that theoretical modelling is not only relevant for simulations but also equally important for experimental activities. Using the measurement of the gravitational acceleration with a plane pendulum as an example, she illustrates the numerous corrections and idealisations that have to be applied after the measurement with the physical apparatus is done. She argues that essentially the model is the device that measures the gravitational acceleration in the experimental setup and concludes, "simulations, as a form of modelling, can fulfil the same role" (Morrison 2009, p. 55). Defenders of this position typically do not claim that simulations are in general epistemically on a par with experiments, but rather argue that this is possible in specific contexts. Massimi and Bhimji (2015) for example do this for the case of the Large Hadron Collider and the discovery of the Higgs boson.

This claim of an equal level of epistemic power only works as long as we stay in the context in which the choice of the model of the target system (cf. Figure 3) has been validated. If we want to understand what happens if a baseball is thrown at a speed close to the speed of light, then our model will not be helpful, because it is not validated for this scenario. We actually know about physical phenomena that would appear in this case that we did not include in our model of the target system (Munroe 2014). We can tie this back to the surprise argument. As long as we use the computer simulation in a context, in which we know the target system very well and do not expect any surprises if we conduct a new experiment, the computer simulation can be granted the same level of trust as an experiment.[28] There are a number of contexts in which the knowledge about the phenomenon is so comprehensive and the computational models have been validated sufficiently to say that the simulation results in this specific context are as trustworthy as experiments would be. In these cases, it does make sense from a practical perspective to call the computer simulation epistemically on a par with experiments.

5 Computer simulations and Beyond the Standard Model searches in high-energy physics

In this Section, I will illustrate that both perspectives discussed above are present in scientific practice. For this, I will describe two examples from high-energy physics research to provide an insight into how computer simulations are used in scientific practice and what epistemic purposes they serve. The examples will revolve around the Standard Model of particle physics (SM) and Beyond the Standard Model (BSM) searches, therefore I will first provide a short description of the SM and the need for BSM searches.

The SM is the theory that describes the elemental particles in the universe and three of the four fundamental forces (all but the gravitational force) governing them. It was developed in the second half of the 20th century and has successfully predicted and explained experimental

[28] We can of course never be completely certain about the results of new experiments; therefore a small possibility of a surprising result in an experiment will remain. This possibility might be deemed epistemically irrelevant at some point, but it underscores the in-principle nature of the surprise argument.

results since then. The discovery of the top quark in 1995 and the tau neutrino in 2000 further confirmed the SM and with the discovery of the Higgs boson in 2012 (ATLAS Collaboration 2012), the last elementary particle of the standard model of particle physics has been found.

At the same time, the SM fails to answer some questions that are central to fundamental physics: Can the fundamental interactions be unified? How did matter originate? What is the nature of dark matter and dark energy? What is the origin of neutrino masses? Why is there more matter than antimatter in the universe? There are also questions concerning the newly found Higgs boson: How to react to the fine-tuning problem of the Higgs mass? What exactly does the Higgs sector look like?

The SM has been tested experimentally to a precision of 10^{-9} in almost all sectors (Mättig 2019), making it one of the best-tested theories that science has seen so far. Models, which are designed to answer the above questions, therefore usually start with the SM and extend it in one way or another. These models are referred to as models Beyond the Standard Model (Lykken 2010). Since the tests of the SM in the energy levels that the LHC could reach, were done extensively and successfully, the extensions or modifications to the SM are often made at energy levels beyond that.

The first example is a part of the simulation infrastructure of the LHC that is used in an area of the SM that is well tested and understood: an event generator as it is used in the ATLAS experiment at the LHC. Two kinds of simulations are essential for the ATLAS experiment: the event generators simulate the collision of two protons and the development of the resulting particles until they reach the detector; the detector simulations simulate the reactions of the real-world detector on the particles that result from the collisions and the following decays.[29] In the experiment the order of the two processes is somewhat turned around. The measurements, which are gathered through the experiment, are the detector signals. They are then used to reconstruct the event that has occurred inside the beam pipe that cannot be observed directly. The simulation of events and their respective detector responses is mainly used to improve the quality of, and the confidence in, this reconstruction. For this there has to be a high confidence that the event generator and the detector simulation produce data that accurately represent the actual events.

[29] A detailed description of the simulation infrastructure at ATLAS can be found at (ATLAS Collaboration 2010). Boge and Zeitnitz (2021) offer a presentation in a philosophical context.

The ATLAS Collaboration has a number of validation procedures to ensure the correctness of the event generators. Systematic tests of the individual components (unit testing) as well as technical tests of their interactions are performed (small-scale and large-scale integration testing). There are three independent frameworks for testing: AtNight, Kit Validation, and the Run Time Tester, which all run on different environments (ATLAS Computing Group 2005). For the validation of the simulations at ATLAS, factorization is crucial, meaning that the validation of the individual steps of the simulation is sufficient for a global validation of the simulation. This factorization is based on theoretical arguments about the physics processes (Mättig 2019). There are different methods to validate the individual steps of the simulation. Results of previous experiments are used together with model assumptions; the simulations are adjusted to data of previous precision measurements. It is also possible to validate the simulation using results of the LHC by using measurements that are complementary to the process that shall be validated, or via in-situ measurements of detector properties (Mättig 2019, p. 645). The combination of these methods, and especially the constant validation against experimental data, allows particle physicists to assign a very high confidence to their simulations, and therefore a very high probability to the interpretations of the measurements, e.g. the discovery of the Higgs boson.

The confidence in the simulation in this example is based on theoretical understanding of the target system (allowing for the factorization argument), extensive verification, and validation against experimental data. In this example, the trust in the simulation is so high that we could call the simulation epistemically on a par with an experiment. Nevertheless, physicists are always careful to assign some remaining systematic uncertainty to the results of the simulation.

The second example is the use of simulations in BSM searches at the LHC. In many cases, a BSM model introduces a new heavy particle that is produced in the collision and decays within the beam pipe, or it predicts a new scattering process or the modification of a SM process. In both cases, one way to test a BSM model is to implement the new particle or process in an event generator and compare the simulation results with detector data. Every new process or particle creates a specific signature in the detector and the BSM model predicts how often this signature should appear.

The most prominent example for the introduction of new particles is supersymmetry (SUSY). The implementation of a new particle or process in an event generator is very time-consuming and most of the time has been spent on SUSY models. SUSY is a principle that proposes that each boson and each fermion has a superpartner in the other group. These partners would be new particles, which have not yet been discovered. One major downside of SUSY is that is has at least 300 free parameters, which means that the model can be fitted to almost every observation and it is very difficult to rule out the concept completely. A further problem is that some processes cannot be separated from SM background processes at the LHC. In these cases, claims about the proposed SUSY process can only be made via the rate of the different processes (Barr et al. 2016).

What is the status of the simulations in these BSM searches? The structure of the event generators is the same as for the SM cases. Let us assume that we have two different event generators for SUSY models. These models are neither validated, nor does data exist that contradicts their predictions. There are no clear arguments why one should prefer one or another model. The only way to decide between different models in this situation is via experimental data. In BSM searches, experiments are therefore clearly epistemically superior to simulations.

BSM searches also serve as an example to illustrate that modelling choices are not directly defined by the target system. In most of the BSM models, very high energies are required to produce a deviation from the SM, and they can therefore not (yet) be empirically tested yet. Physicist have very different expectations and preferences concerning BSM models. BSM models with supersymmetry are the most popular ones, but many other concepts are pursued as well (Chall et al. 2019). Asked for the criteria regarding their model preferences, “elegance” and “naturalness” were the most popular answers. Although all of these physicists have the same target system in mind (the interaction of particles), they choose different equations or entities for their models of the target system (Mättig and Stöltzner 2019).

6 Epistemically relevant differences between computer simulations and experiments

Both the arguments in Sections 3 and 4 and the examples in Section 5 are not leading towards a completely clear conclusion about the epistemic status of computer simulations. There are certain contexts in which an experiment provides epistemic work that a computer simulation could not, there are contexts in which both simulations and experiments can be used, and there are contexts in which the use of a simulation could be epistemically advantageous to the use of an experiment. In all these cases it depends on the context, on the question that has to be answered, and on the experiences that have already been made in the domain in question. A conclusion that makes a general statement about the epistemic hierarchy between computer simulations and experiments would not do justice to these different contexts and to the different epistemic tasks that experiments and computer simulations can fulfil. I therefore suggest that it is more useful to consider different epistemic tasks that can be fulfilled by both experiments and computer simulations and evaluate computer simulations and experiments in each concrete context based on their respective usefulness for each of these tasks.

I suggest three concrete formulations that describe aspects of the use of computer simulations or experiments that are epistemically relevant and connect them to the different perspectives and arguments. Further aspects are imaginable.

Epistemic Aspect 1: The necessity of the method for a certain goal

One aspect is the necessity of a method for a certain goal. If Method A is necessary to acquire knowledge about a certain phenomenon, but Method B is not, then Method A has a greater epistemic value in this setting. If both methods are necessary, nothing can be said about their hierarchy in terms of epistemic value in respect to this aspect.

Both Morrison (2015) and Massimi and Bhimji (2015) use the argument, that simulations are necessary for the discovery of the Higgs-boson and should therefore be understood as epistemically on a par with experiments. Since none of them deny that experiments are also necessary, this argument does not hold. One cannot make conclusions about the epistemic value of Method A in respect to Method B from the fact that both are necessary for a certain task.[30]

[30] Apart from that, it is debated whether computer simulations were actually necessary for the discovery of the Higgs-boson (Gregor Schiemann, Christian Zeitnitz and Michael Krämer, personal communication 2019).

In the context of the methodological perspective, the necessity of experiments for certain tasks is an argument for their epistemic privilege, as there are many contexts in which one can argue that an experiment is necessary to acquire knowledge. This is especially relevant in areas which have not been investigated, or not been sufficiently investigated in an empirical way. There are, however, also contexts in which a simulation is the only method that enables the scientist to acquire certain knowledge because a domain in question is not experimentally accessible or for all practical purposes the amount of calculation needed requires the use of a computer.

Epistemic Aspect 2: Discovery potential

A second epistemic aspect of a method is the potential for the discovery of new knowledge through this method. If Method A has a higher potential to discover new knowledge than Method B, then Method A has a greater epistemic value.

This aspect, in connection with the surprise argument, can be used to argue for the epistemic superiority of experiments, since experiments can produce completely new and unexpected results in a way computer simulations cannot. The discovery potential of experiments is therefore one of the main arguments in the methodological perspective.

There are, however, also contexts in which it can be claimed, that computer simulations have a greater potential to discover new knowledge than experiments simply in virtue of the fact that a computer simulation can be performed while no comparable experiment is possible. In these contexts, it can be argued that computer simulations have an epistemic advantage over experiments. In practice this seems to be the case in some scientific disciplines. In astrophysics for example, the epistemic status of simulations is a lot higher simply because experiments are often not possible. Apart from that, a special kind of models, namely targetless fictional models, can enable us to learn something about situations that do not, and maybe cannot, occur in the real world, and are therefore not accessible via experiments, for example: three sex populations, Norton's dome, a world without gravity.[31] One could show that a phenomenon is not stable in a system by creating it in a model world and showing that it

[31] Michela Massimi presented this in her keynote lecture "What Scientific Models Are For" at the "Models and Simulations 8" conference at the University of South Carolina, March 2018.

dissolves. In a setting like this, one can gain knowledge via the simulation that one could never gain via an experiment.

Epistemic Aspect 3: The degree of trust in the results of the respective methods

If two methods are concerned with exactly the same phenomenon and both methods are used, the question of trust in the method can be reduced to the question of whether we trust Method A or Method B in case the two methods have different results. Alternatively, a situation can arise where the trust in the result of Method A is so high, that there is no need to employ Method B as well.

In case of disagreement between the results of an experiment and a computer simulation, the scientific practice clearly favours the experiment. The trust in simulation results is based on the validation through experimental results ("epistemic priority") and therefore experimental results that disagree with the results of computer simulations lead to doubt about the correctness of the computer simulation. In principle, this situation of disagreement can always occur, no matter how rigorously the computer simulation has been validated with experimental results beforehand. The likelihood of this situation occurring does become very small, however, if the simulated phenomenon is very well known and understood, and the computer simulation has been extensively validated and successfully used. In this situation scientists trust the results of a computer simulation to a degree that there is no need to perform an experiment as well.

7 Conclusion

In this paper I have offered a reconstruction of the modelling and testing process of a computer simulation. Using the insights from this reconstruction I have structured and evaluated the arguments in the debate about the epistemic hierarchy between computer simulations and experiments and identified two perspectives. These two perspectives, the methodological and the pragmatic perspective on what it means for an experiment and a computer simulation to be epistemologically on a par, can explain why different authors think differently about the epistemic hierarchy of computer simulations and experiments. Both perspectives have their merits, and having both perspectives available is helpful for a good understanding of the relationship between computer simulations and experiments. The pragmatic perspective cannot be generalised and a sufficient validation of the model for one specific context has to be proven to treat the simulation as epistemologically on a par.

The term ‘epistemologically on a par’ itself is problematic, because epistemically relevant aspects are subsumed under it and consequently not addressed in enough detail. I therefore suggest discarding the term ‘epistemologically on a par’ and rather explicate epistemologically relevant aspects. Some of these aspects, which are used in the debate about epistemic hierarchy, but are not discussed explicitly, are: the necessity of a method for a certain goal, the discovery potential, the degree of trust in the results of a method.

References

ATLAS Collaboration. 2010. “The ATLAS Simulation Infrastructure”. The European Physical Journal C 70, no. 3: 823–74. doi:10.1140/epjc/s10052-010-1429-9.

ATLAS Collaboration. 2012. “Observation of a new particle in the search for the Standard Model Higgs boson with the ATLAS detector at the LHC”. *Physics Letters B* 716, no. 1: 1–29. doi:10.1016/j.physletb.2012.08.020.

ATLAS Computing Group. “Computing technical design report. Technical report, CERN.”, last modified 2005. https://cds.cern.ch/record/837738/files/lhcc-2005-022.pdf.

Barr, Giles, Robin Devenish, Roman Walczak, and Tony Weidberg. 2016. *Particle physics in the LHC era*. Oxford: Oxford University Press.

Beisbart, Claus. 2012. “How can computer simulations produce new knowledge?”. *European Journal for Philosophy of Science* 2, no. 3: 395–434. https://doi.org/10.1007/s13194-012-0049-7.

Boge, Florian J. 2019. “Why computer simulations are not inferences, and in what sense they are experiments”. *European Journal for Philosophy of Science* 9, no. 1: 13. https://doi.org/10.1007/s13194-018-0239-z.

Boge, F. & Grünke, P. forthcoming “Computer simulations, machine learning and the Laplacean demon: Opacity in the case of high energy physics.” forthcoming in Resch, Kaminski, and Gehring (Eds.), *The Science and Art of Simulation II,* Springer. Preprint available: http://philsci-archive.pitt.edu/17637/

Boge, Florian. J. and Zeitnitz, Christian. 2021. Polycratic hierarchies and networks: What simulation-modeling at the LHC can teach us about the epistemology of simulation. *Synthese*.

Chall, Cristin, King, Martin, Mättig, Peter, and Stöltzner, Michael. 2019. “From a Boson to the Standard Model Higgs: A Case Study in Confirmation and Model Dynamics”. *Synthese*. https://doi.org/10.1007/s11229-019-02216-7

Franklin, Allan. 1986. *The neglect of experiment*. Cambridge: Cambridge Univ. Press.

Frigg, Roman, and James Nguyen. 2017. “Models and Representation”. In *Springer Handbook of Model-Based Science*, edited by Lorenzo Magnani and Tommaso Bertolotti, 49–102. Cham: Springer International Publishing.

Galison, Peter, 1987. *How experiments end*. Chicago: University of Chicago Press.

Galison, Peter. 1996. “Computer Simulations and the Trading Zone”, in *The Disunity of Science: Boundaries, Contexts, and Power*, edited by Peter Galison and David Stump. Stanford: Stanford University Press

Gramelsberger, Gabriele. 2010. *Computerexperimente: Zum Wandel der Wissenschaft im Zeitalter des Computers*. s.l.: transcript Verlag.

Guala, Francesco. 2002. “Models, simulations, and experiments”. In *Model-Based Reasoning: Science, Technology, Values*, edited by Lorenzo Magnani and Nancy J. Nersessian, 59–74. Boston, MA, s.l.: Springer US.

Hacking, Ian. 1983. *Representing and intervening: Introductory topics in the philosophy of natural science*. Cambridge: Cambridge Univ. Press. http://dx.doi.org/10.1017/CBO9780511814563.

Harré, Rom. 2003. “The materiality of instruments in a metaphysics for experiments”. In *The philosophy of scientific experimentation*, edited by Hans Radder, 19–38. Pittsburgh, Pa: University of Pittsburgh Press.

Humphreys, Paul. 2004. *Extending Ourselves: Computational Science, Empiricism, and Scientific Method*. New York: Oxford University Press.

Humphreys, Paul. 2009. “The philosophical novelty of computer simulation methods”. *Synthese* 169, no. 3: 615–26. doi:10.1007/s11229-008-9435-2.

Lykken, Joseph D. 2010 “Beyond the standard model”. *arXiv preprint arXiv:1005.1676*.

Massimi, Michela, and Wahid Bhimji. 2015. “Computer simulations and experiments: The case of the Higgs boson”. *Studies in History and Philosophy of Science Part B: Studies in History and Philosophy of Modern Physics* 51: 71–81.

Mättig, Peter. 2019. “Validation of Particle Physics Simulation”. In *Computer Simulation Validation: Fundamental Concepts, Methodological Frameworks, and Philosophical Perspectives*, edited by Claus Beisbart and Nicole J. Saam, 631–60.

Mättig, Peter, and Michael Stöltzner. 2019. “Model choice and crucial tests. On the empirical epistemology of the Higgs discovery”. *Studies in History and Philosophy of Science Part B: Studies in History and Philosophy of Modern Physics* 65: 73–96. doi:10.1016/j.shpsb.2018.09.001.

Miller, Ron. 2013. “Using Computer Simulation: In Search of the Perfect Curve Ball”. Accessed November 15, 2019. https://smartbear.de/blog/test-and-monitor/using-computer-simulation-in-search-of-the-perfect/?l=ua.

Morgan, Mary S. 2005. “Experiments versus models: New phenomena, inference and surprise”. *Journal of Economic Methodology* 12, no. 2: 317–29. doi:10.1080/13501780500086313.

Morrison, Margaret. 2009. “Models, measurement and computer simulation: The changing face of experimentation”. *Philosophical Studies* 143, no. 1: 33–57. doi:10.1007/s11098-008-9317-y.

Morrison, Margaret. 2015. *Reconstructing Reality: Models, Mathematics, and Simulations*. New York: Oxford University Press.

Munroe, Randall. 2014. *What if? Serious scientific answers to absurd hypothetical questions*. Boston, Mass.: Houghton Mifflin Harcourt.

Oberkampf, William L., and Matthew F. Barone. 2006. “Measures of agreement between computation and experiment: Validation metrics”. *Journal of Computational Physics* 217, no. 1: 5–36. doi:10.1016/j.jcp.2006.03.037.

Oberkampf, William L., and Christopher J. Roy. 2012. *Verification and validation in scientific computing*. Cambridge: Cambridge Univ. Press.

Oberkampf, William L., Timothy G. Trucano, and Charles Hirsch. 2004. “Verification, validation, and predictive capability in computational engineering and physics”. *Applied Mechanics Reviews* 57, no. 5: 345–84. doi:10.1115/1.1767847.

Oestreicher, Christian. 2007. "A history of chaos theory." *Dialogues in clinical neuroscience* 9.3: 279.

Oreskes, N., K. Shrader-Frechette, and K. Belitz. 1994. “Verification, validation, and confirmation of numerical models in the Earth sciences”. *Science (New York, N.Y.)* 263, no. 5147: 641–46. doi:10.1126/science.263.5147.641.

Parke, Emily C. 2014. “Experiments, Simulations, and Epistemic Privilege”. *Philosophy of Science* 81, no. 4: 516–36. doi:10.1086/677956.

Parker, Wendy S. 2009. “Does matter really matter? Computer simulations, experiments, and materiality”. *Synthese* 169, no. 3: 483–96. https://doi.org/10.1007/s11229-008-9434-3.

Roy, Christopher J. 2005. “Review of code and solution verification procedures for computational simulation”. *Journal of Computational Physics* 205, no. 1: 131–56. doi:10.1016/j.jcp.2004.10.036.

Sargent, R. G. 2013. “Verification and validation of simulation models”. *Journal of Simulation*, 7: 12–24. doi:10.1057/jos.2012.20.

Winsberg, Eric. 2009. “A tale of two methods”. *Synthese* 169, no. 3: 575–92. https://doi.org/10.1007/s11229-008-9437-0.

Winsberg, Eric B. 2010. *Science in the age of computer simulation*. Chicago, London: University of Chicago Press.

Chapter 3: How do we learn from Computer Simulations? A Critical Examination of the "Argument View" on Simulation

Abstract

The epistemic characteristics of computer simulations (CSs) in practices of scientific knowledge acquisition have given rise to lively debates in philosophy of science. In contrast to the prominent claim that with regard to epistemic characteristics CSs are comparable to ordinary experiments, Beisbart has recently proposed that CSs can be reconstructed as arguments, which fully capture their epistemic power. In this paper, we critically examine this "argument view" on CS and argue that while it does indeed capture some significant characteristics of CSs, its answer to the question of how CSs can contribute to the acquisition of knowledge remains unsatisfying. Nevertheless, we show that critically discussing the argument view and its limitations offers fruitful insights on two central topics in the philosophical literature on computer simulations: their (allegedly) opaque character and the role of inductive reasoning in the context of CSs.

1 Introduction

Over the last several decades, computer simulations (CSs) have increasingly gained a foothold in numerous scientific areas. Given that they have shaped and continue to shape practices of knowledge acquisition in various respects (e.g., in the climate sciences or materials sciences), it is not very surprising that their epistemological characteristics have sparked lively debates. Many philosophers have addressed questions concerning the epistemology of CSs by pointing out their similarities to or differences from experiments (see, e.g., Massimi und Bhimji 2015; Morrison 2009; Morgan 2005; Parker 2009).

A different approach, however, has been suggested by Claus Beisbart (2012; see also Beisbart and Norton 2012). Rather than locating the epistemological characteristics of CSs in proximity to those of experiments, Beisbart proposes that in terms of the way in which they enable acquisition of new knowledge CSs are comparable to deductive arguments. Not only, so

Beisbart (2012, 403) argues, can they be reconstructed as arguments, they also have the same epistemic power as arguments.

Considering the deterministic character and the algorithmic structure of CSs, comparing them to arguments in terms of epistemology certainly has some immediate appeal – and unsurprisingly, Beisbart is not the only scholar to emphasize the inferential nature of CSs. For example, Winsberg (1999, 275) points out that CSs "involve a complex chain of inferences that serve to transform theoretical structures into specific concrete knowledge of physical systems", while Stöckler (2000, 369) explicitly states that "[s]imulations are arguments, not experiences". Moreover, examining the epistemic role of models in science more broadly, Kuorikoski and Lehtinen (2009) argue that "the epistemic reach of modelling is precisely the same as that of argumentation" (122) and that regarding models as "surrogates" for target systems on which we experimentally intervene may even lead us to overestimate their epistemic power (125). Finally, Kuorikoski (2012, 175) states that "in principle, every simulation run is just a long deduction".

However, while there is clearly something to be said for the comparison between arguments and CSs, does this comparison really answer the question of how CSs enable us to gain new knowledge? Florian Boge (2019, 12), for example, points out that while CSs – given their "algorithmic nature and the implied rigidity of the input-output relation" – are certainly "respecting the logics of arguments", they clearly differ in their pragmatics. Moreover, even Beisbart (2012, 417) himself admits that in certain respects, the argument view abstracts from relevant epistemological issues.

In this paper, we discuss the argument view and its limitations in order to examine how and to what extent drawing parallels between deductive argumentation and simulations can help to clarify the epistemological characteristics of CSs.

Our assessment is two-sided. On the one hand, we agree with Beisbart that the relationship between simulation input and output can fruitfully be compared to the relationship between the premises and conclusion of a deductive argument. Moreover, we argue that by taking this very circumstance seriously, one can gain insight on a prominent topic of discussion in the literature on CSs, namely the oft-asserted epistemic opacity of simulations.

On the other hand, however, we do not fully agree with Beisbart’s statement that the *epistemic power* of CSs is that of deductive arguments. Although we grant that simulation outputs are generated by means of deductive inference, acquiring knowledge with the help of simulation often requires adopting practices of inductive reasoning, such as when a wide variety of “raw” simulation outputs is referred to in order to make claims about a general pattern or rule of behavior. Here, the analogy between arguments and simulations appears to falter – and promoting an “argument view” on CSs may even mislead us into overlooking the inductive leap that exists in simulation-based practices of reasoning.

Ultimately, we argue that Beisbart’s suggestion of comparing CSs to arguments is clearly enriching when it comes to an understanding of their general functioning and the process of output generation. The comparison does have its limits, however, and cannot comprehensively explain how new knowledge can be obtained by the use of a CS.

2 The argument view on CS

As motivation for adopting an argument perspective on simulation, Beisbart (2012, 396) observes that it is puzzling that computer simulations allow scientists to acquire knowledge about physical systems without directly observing them. He argues that despite CSs’ manifold applications in various fields of research and their apparent success in contributing to scientific knowledge acquisition, it is no trivial matter to explain *how* computer simulations enable scientists to obtain knowledge about physical targets. According to Beisbart (396), we face the following question (Q_CS) “How can scientists gain new knowledge by running computer simulations?”

Starting from the idea that CSs bear a pre-theoretical similarity to thought experiments and taking into consideration Norton’s (2004) argument view on thought experiments, Beisbart (2012, 401) proposes that CSs enable acquisition of new knowledge because they are arguments:[32] since it is “uncontroversial that we can obtain new knowledge by running

[32] One may object that in the case of deductive arguments, the conclusions are already in some sense “entailed” in the premises and that, therefore, the idea that CSs do their epistemic work in the same way as arguments does not show how they contribute to the acquisition of new knowledge. Yet, with examples from mathematical argumentation in mind, we grant to Beisbart that it seems largely uncontroversial that elaborate deductive inferences can lead to new insights.

through an argument", identifying CSs with deductive arguments would provide an answer to (Q_CS).

Beisbart (403) states that the argument view on CS consists of two claims, namely that (I) "each computer simulation can be reconstructed as an argument such that the epistemic power of the CS is that of the argument" and (II) that running a simulation is to execute the reconstructing argument. Boge (2019) has convincingly pointed out that the first claim should more appropriately be divided into two distinct claims, so that we can reconstruct the argument view as consisting of three central claims, namely:

(1) that CSs can be reconstructed as arguments (R_CS),

(2) that their epistemic power is that of the reconstructed argument (EP_CS) and

(3) that running a CS is to execute the argument (Ex_CS).

Before we examine whether computer simulations can convincingly be compared to arguments, let us elucidate what precisely is meant by 'computer simulations' in the context of Beisbart's view. He points out that there are at least two ways of speaking about CSs:

> In a narrow sense, to run a computer simulation is to run a suitable simulation program. In a broader sense, we may […] think of programming, testing the program and evaluating the results as part of a CS too. (Beisbart 2012, 398)

Beisbart explicitly states that within the scope of the argument view, CSs are referred to in the narrower sense. Following Parker (2009, 488), he suggests using the term 'computer simulation study' as a broader overarching notion for various activities involved in the scientific usage of simulation programs such as the examination of how the outputs of a simulation process are affected if several runs are performed with different parameters. Interestingly, he claims that the argument view *abstracts* from such activities and focuses solely on the *run* of simulation programs (Beisbart 2012, 398).

Against this backdrop, it seems that the central motivation for comparing CSs (in the aforementioned narrow sense) with arguments is the idea that the relationship between the inputs and results of simulation runs bears a certain similarity to the relationship between the premises and conclusions of deductive arguments. In the following two sections, we will further clarify this supposed similarity as well as Beisbart's claim that the epistemic power of CSs is that of the reconstructing argument.

2.1 The reconstruction thesis

In order to illustrate the idea that computer simulations can be reconstructed as arguments, Beisbart (2012, 404ff.) discusses the example[33] of a simulation of the (approximated) behavior of a driven and damped pendulum based on an implementation of the Euler method.[34]

Structurally, the implemented algorithm of the Euler method consists of two assignment statements (which set the values of two respective variables that specify the initial conditions of the pendulum) and a foreach loop (which specifies how the values of these two variables change over *n* iterations).

As Beisbart (406ff.) argues, the simulation program can be reconstructed as a deductively valid[35] argument about the position of the pendulum after a certain number of time steps. According to Beisbart, the 'simulation argument' consists of four premises, with two premises specifying (1) the initial angular position φ_0 and (2) the initial angular velocity Vφ_0 of the pendulum at a certain time point, and two further premises (3, 4) specifying how the values of these two characteristics at each time point depend on their value at a respectively earlier time point. Depending on whether one focusses on a *single run* of the simulation with concretely specified initial values or on the general structure of the simulation program, Beisbart suggests conceiving of CSs as arguments or argument schemes:

> The idea is that φ_0 and $V\varphi_0$ are place-holders for values of the physical characteristics in a suitable coordinate frame. We obtain different arguments (or argument schemes) if we insert different values. This then is the picture we have come up with: The execution of an algorithm by a computer simulation program using concrete input values can be reconstructed using an argument or an argument scheme, depending on

[33] See Boge (2019) for another reconstruction based on an example from high-energy physics.

[34] In the simulation model, the second-order differential equation that describes the motion of the pendulum is replaced by two coupled first-order equations which are, in turn, approximated by two difference equations (with time being discretized).

[35] Beisbart explicitly states that CSs are *deductive* arguments rather than, for example, inferences to the best explanation: "If we abstract from roundoff and truncation errors and the arguments used in validation, CSs are deductive arguments" (2012, 427).

> how the scientist thinks of the simulation. [...] I will for simplicity assume that the single run is properly reconstructed in terms of one argument rather than an argument scheme. (409)

Through the simulation process, the argument that describes the trajectory of the pendulum is executed. Accordingly, the results of the simulation run constitute the conclusion of the simulation argument.

2.2 The epistemic power thesis

Given the aim of clarifying the role of CSs in contexts of knowledge acquisition, the heart of the argument view consists in the idea that computer simulations can not only be *reconstructed* as arguments ([R_CS]), but also in the notion that they have the same epistemic power as the reconstructing argument ([EP_CS]). For Beisbart (2012, 404), the second thesis seems to be directly implied by the first: as he argues, “the reconstructions fully capture the epistemic power of the underlying computer simulations.” But what precisely does it mean for CSs to be ‘epistemically powered’ in a certain way?

Let us approach this question by clarifying the success criteria that Beisbart formulates with respect to his comparison between the epistemic power of CSs and arguments. According to Beisbart, we can determine that the epistemic power of a CS is that of the reconstructing argument if we show that one of the following two statements is true: (1) the reconstructing argument captures all epistemically relevant aspects of the CS, or (2) “[all] epistemic work that a computer simulation does could also be done by executing the reconstructing argument without computers” (416).

Regarding the first claim, Beisbart argues that it is implausible to assume that anything of epistemic relevance is lost when CSs are reconstructed as arguments. Given that CSs perform computations that are usually taken to “trace the values of empirical characteristics” of some physical system and therefore can be ‘translated’ into empirically substantial statements[36], and assuming that the process of result generation can be understood as a deductive

[36] We will address this claim about translatability in more detail below.

inference, nothing that is epistemically relevant seems to be lost when reconstructing CSs as arguments (416).

Looking at the second claim, Beisbart admits that a significant challenge lies in the fact that, even if computer simulations can indeed be reconstructed as arguments, the premises of these simulation arguments often turn out to be false because they are based upon approximated, idealized or abstracted assumptions. If the epistemic power of arguments stems from their capacity to inferentially transform prior knowledge and if CSs contribute to knowledge acquisition *in the same way* as arguments – how can their epistemic power be captured in this way when they are, in many cases, built upon false premises (418)?

According to Beisbart, we can maintain (EP_CS) if we grant that the argument view is limited to explaining how scientists can acquire knowledge about *models*:

> Admittedly, the argument view does not explain why scientists are justified to transfer knowledge from the model to the target. Additional considerations are necessary to justify this inference. But it should be possible to present these considerations as arguments too. Further, even if it is not, this does not mean that the argument view stumbles. The argument view can explain how scientists gain new knowledge about a model. It can thus explain how computer simulations make a contribution to new model-based knowledge about the real world. (419)

Two points are crucial here: Firstly, Beisbart emphasizes that 'justificatory considerations' typically have the form of arguments, too. Thus, the fact that we require additional considerations with regard to what is usually called the *validation* of a CS does not preclude comparing CSs to arguments with respect to their epistemic power. Secondly, the key idea of (EP_CS) seems to be that by regarding CSs as arguments, we can – even if we do not capture all activities involved in working with CS – learn *how* they can act as a means of scientific knowledge acquisition in general: even if we abstract from the fact that additional validatory activities need to be performed in order to ensure that CS can help to acquire new knowledge about the *world* rather than only about a model, the argument view still teaches us about the general *means* by which new insights are acquired.

Overall, it seems that the argument view is committed to the idea that the inferential transformation of CS inputs is the source of epistemic gains. In the case of ordinary arguments, according to Beisbart, we can learn something new about the world by drawing on known premises and transforming these in a truth-preserving manner. We would usually consider arguments successful if they are sound, i.e., if their premises are true and all inferential steps valid. This process appears to be structurally similar in the case of CS: “If the […] premises are known to be correct, we obtain a sound argument the conclusion of which may represent new knowledge. Using this argument, we can explain how the simulations produce new knowledge” (Beisbart 2012, 419).

In their 2012 paper, Norton and Beisbart emphasize even more pointedly their idea that while ordinary experiments are epistemically powered by *discovery*, CSs are powered by *inference*: While in experimental contexts we prepare and manipulate an apparatus and “observe the world to learn about it,” CSs contribute to the acquisition of knowledge by referring to prior knowledge and enabling us to *draw inferences* based on this knowledge (Beisbart und Norton 2012, 9). Moreover, they argue that while discovery-based knowledge acquisition practices “[go] directly to the world” (9), CSs provide new knowledge in that they inferentially *transform* known premises rather than relying on an interaction with the world.

In summary, the central ideas of (EP_CS) seem to be that (1) to advocate an argument view on CS is to emphasize that new knowledge is generated inferentially rather than by means of discovery and that (2) the comparison of the epistemic power of CSs to arguments asserts that we learn from CSs and from arguments in a generally similar way.

3 Can CSs be reconstructed as arguments?

Beisbart is aware that his suggestion of reconstructing CSs as arguments raises (at least) two immediate doubts: How can a run of a simulation be reconstructed as an argument if “what the computer program takes as input are of course mere numbers” (2012, 407)? And how are we entitled to say that the conclusion deductively follows from the premises given that computers will produce roundoff errors?

Addressing the first concern, Beisbart (407) specifies that while the premises of the simulation argument are *represented* as numbers, they should at the same time be empirically meaningful for the working scientist. Thus, if we want to maintain that simulations can be reconstructed as arguments, we need to show that their premises can be ‘translated’ into statements with empirical meaning, e.g., statements about properties of a physical system or about the way in which various such properties are connected. According to Beisbart, it can easily be shown that this "condition of translatability" is met: for the scientist who runs a CS, the initial conditions and algorithmic statements of a given computer program "encode the values of physical characteristics of the system in certain units" (407).

In the example discussed above of the CS of a pendulum, the variable initialization 'set phi[0] = $\varphi_{t=0}$' (with φ_t referring to the angle between the thread and a plumb line at time *t*) can be regarded as a premise, according to Beisbart, in that it can be interpreted as a statement about the value of the initial angular position of the pendulum after zero time steps. During the simulation process, the value of φ is updated according to the transformation rules (which, as indicated above, themselves constitute premises of the simulation argument). Although the results of the pendulum simulation will take the form of numerical values, we would generally understand these values as denoting physical characteristics of the system under investigation, namely the position of the pendulum at different time steps expressed in suitable coordinates, i.e., its trajectory.

Looking at the second objection, however, Beisbart admits that the reconstruction thesis has to be refined in order to take into account the fact that the results delivered by a computer program will differ from the results that would have been provided if the computational process had followed the algorithms and transformed the original numerical inputs in an *exact* way.

Naturally, computational processes will produce roundoff errors or truncation errors. Thus, in order to maintain the idea that simulations can be reconstructed as deductively valid arguments, Beisbart (411f.) suggests that the reconstructing argument and its conclusion need to be modified so as to account for the circumstance that the algorithmic rules are followed only *approximately*. For example, the premise of the 'simulation argument' that specifies how the angular velocity of the pendulum changes over *n* iterations must be refined such that it entails that the angular velocity of the pendulum at a given time step will *approximately* take a certain value.

Beisbart defends his view that the reconstruction thesis can be upheld despite this modification. He argues that scientists are usually not interested "in the exact numbers" but in making general statements about the behavior of the system which the simulation represents (411).

Overall, we agree with this consideration and with the general idea that the relationship between simulation output on the one hand and initial conditions and algorithms on the other can profitably be compared to the relationship between (deductively derived) conclusions and their premises. As Boge (2019, 13) has pointed out, "[the] evidenced possibility of reconstruction in terms of arguments speaks in favor of CSs respecting the logic of arguments. This may be understood perfectly well in terms of their algorithmic nature and the implied rigidity of the input-output relation, similar to truth preservation in deductively valid arguments."

4 Epistemic opacity and the inferential character of CSs

It is often claimed that despite their central role in studying the behavior of complex systems, among other things, CSs have the disadvantage of being epistemically opaque. In recent debates, much has been said about the sources of this opacity as well as about the ways in which the opaque character of CS may constitute a threat to our epistemic ends (see, e.g., Humphreys 2004, 2009; Kaminski 2018; Lenhard 2006, 2019). Before we proceed with an examination of the epistemic power thesis, let us first elaborate on how the idea that CSs can be reconstructed as arguments might contribute to a better understanding of why and in what respect the oft-stated opaque character of complex computing systems is of epistemological importance.

A prominent definition of epistemic opacity has been proposed by Humphreys:

> A process is epistemically opaque relative to a cognitive agent X at time t just in case X does not know at t all of the epistemically relevant elements of the process. (2009, 618)

Which elements should be considered "epistemically relevant" remains an open question. In the strictest interpretation, every computational step of a simulation process is epistemically relevant since an error in any of the steps could corrupt the simulation result. In the case of CSs, it seems that Humphreys assumes that opacity in fact stems from a lack of access to individual steps of the simulation and from the inability to examine how specific elements of the computational process affect the results. As he puts it, the "dynamic relationship between the initial and final states of the core simulation" is in many cases epistemically opaque because "most steps in the process are not open to direct inspection and verification" (Humphreys 2004, 147f.).

Against this backdrop, CSs seem to differ clearly from 'ordinary' arguments: one would normally say that arguments are epistemically transparent in the sense that the steps involved in inferring the respective conclusions are accessible or traceable by cognitive agents. In the case of CSs, however, it seems that due to their complex dynamics or the sheer number of computational steps involved, such transparency usually cannot be achieved.

Responding to the potential concern that the epistemic opacity of some CSs renders it untenable to maintain that they are arguments, Beisbart and Norton (2012, 416) simply state that "[there] is no requirement that an argument be humanly comprehensible. All that is required is that it is a sequence of propositions conforming to the rules of the applicable logic." By definition, the soundness of an argument does not depend on anything other than the validity of the inferences it contains and the truth of its premises. Does this, however, imply that epistemic opacity should be of no concern?

In order to better identify the challenges related to the non-traceability or inaccessibility of the steps that occur when computing systems enter the stage of scientific knowledge acquisition, let us take a brief look of the widely discussed historical example of Haken and Appel's computer-assisted mathematical proof of the four color theorem. The four color theorem (4CT) is a famous theorem in graph theory stating that only four colors are required to color any map such that no two adjacent regions have the same color. In 1976, Kenneth Appel and Wolfgang Haken announced their proof of this theorem, which was based on the idea of reducing all possible map configurations to a limited number of configurations, each of which could then be verified separately (see Appel and Haken 1989). The proof was, however, contested by mathematicians among other things because of its extensive reliance on computer assistance[37] and because of the fact that it was impossible or at least infeasible for humans to verify all the steps involved in it.[38]

The example of 4CT is interesting for two reasons. Firstly, it was the first time that a mathematical proof was performed with extensive computer assistance and therefore raised questions about the status of computer-generated deductions. Secondly, mathematical proofs are prime examples of deductive reasoning. Since we agree with Beisbart that simulation processes can be reconstructed as deductive arguments, the reservations about the proof of 4CT are relevant to computer simulations as well.

In his discussion of the philosophical implications of Appel and Haken's computer-assisted proof of 4CT, Tymoczko (1979) points to an interesting ambivalence: on the one hand, by and large, most mathematicians seem to have considered Appel and Haken's computer-assisted approach to be reliable enough to show that the 4CT is correct; on the other hand, many have emphasized that the proof does not adhere to the same standards as 'ordinary' mathematical proofs, because "[i]t has not been checked by mathematicians, step by step, as all other proofs have been checked. Indeed, it cannot be checked that way." (70)

Interestingly enough, Tymoczko suggests that it is not the chosen mathematical approach used in Appel and Haken's proof that is in itself opaque or of questionable reliability: "There is a very clear idea of what the computer is supposed to be doing – we have a good understanding of reduction techniques" (74). Discussing the question of how the reliability of the computer-assisted proof differs from that of 'traditional' proofs, he states:

> The reliability of the machine is ultimately a matter for engineering and physics to assess. Of course, even if we grant that the machine does what it is supposed to – follow the program – there remains the question of whether the program does what *it* is supposed to. This question can be difficult to answer. The task of evaluating programs is a topic of computer science, but at present there are no general methods for accomplishing it at this level. Programs themselves are written in special

[37] Appel and Haken's proof depends on the existence of reducible configurations of maps. The main role of the computer was to check each of the 1936 relevant configurations for reducibility, which took over 1000 hours of computing time.

[38] By now a simpler proof with the same general approach has been found and the theorem is widely accepted as proven.

'languages', and many of them can be quite complex. They can contain 'bugs' or flaws that go unnoticed for a long time. The reliability of any appeal to computers must ultimately rest on such diffuse grounds as these. (74)

Tymoczko's insightful assessment indicates some differences between reservations about the 4CT proof on the one hand and reservations that scientists may have when confronted with a new 'traditional' proof on the other hand. Traditionally, a central reason for demanding that each step of a mathematical proof be verified is that an inferential misstep might have occurred at some point.

In the case of inferences drawn by means of computers, it seems that, on the one hand, one could argue that *as soon as* (I) the premises of a simulation program are regarded as accurate and (II) computational processes are assessed as being sufficiently reliable, there is no need to verify all inferential steps individually (and this seems to be what underlies Beisbart and Norton's statement mentioned above). One might even stress, perhaps, that computers are usually inferentially *more* reliable than human agents due to their strict adherence to algorithmic rules (consider for example that often we use pocket calculators precisely to *confirm* 'manual' calculations).

On the other hand, it seems that the crucial challenge of working with sophisticated computing systems is that the very practice of determining *if* (I) and (II) are fulfilled becomes much more difficult. As Tymoczko's considerations suggest, a central element of the reservations about the 4CT proof arose due to the concern that programming languages may contain bugs or that hardware malfunctions may be present.

What can we learn from this? The example of 4CT and scientists' discomfort with its computational proof indicates that the mere fact that some computationally-assisted deductions are too difficult for a human agent to perform may not *inherently* be an epistemic defect. Rather, one important elephant in the room seems to be that relying on complex computing systems crucially involves dealing with *uncertainty*: the lack of observability of the 4CT proof seems to constitute a challenge not 'in itself' but in that – due to its complexity and the sheer number of computations involved – it is difficult to estimate whether and where potential sources of errors occur. As Tymoczko (1979, 74) puts it, the reliability of the 4CT proof "rests on the assessment of a complex set of empirical factors."

Let us now apply these findings to the case of CSs and to the issue of epistemic opacity. In the case of the 4CT, it seems that scientists’ discomfort was not primarily about the fact that the deduction was too lengthy for human agents to perform, but had to do, inter alia, with uncertainty about *whether* the deduction had been performed as intended. This seems important for the case of CSs. By accepting the reconstruction thesis and acknowledging the algorithmic nature of CSs, we have reason to claim that once we are certain enough *that* the reconstructing argument is actually deduced, it is not necessary to directly inspect and verify each step of the process. At the same time, such certainty is not always easy to attain as CSs are often quite complex. This, however, suggests that the worries underlying the diagnosis that computer simulations are epistemically opaque might be fruitfully addressed not primarily by striving for an investigation of single deduction steps but by performing suitable practices of error and uncertainty management (Newman 2016; see also Kroll 2018).

5 Going beyond the reconstructing thesis

Thus far, we have suggested, in accordance with Beisbart, that the relationship between simulation input and output can be compared to the relationship between the premises and conclusions of deductive arguments. Moreover, we have outlined how reconstructing CSs as arguments can help shed light on the often-asserted challenge of epistemic opacity.

However, recall that Beisbart’s claim goes further: not only can CSs be reconstructed as arguments, the argument view can furthermore illuminate how CSs enable scientists to learn something new about the world. We will argue that while the reconstruction thesis can be maintained, the situation is different when it comes to this second aspect.

Our examination consists of two parts. First, we focus on the validation of CSs. Even though Beisbart admits that the argument view remains largely silent about validation, he seems to suggest that setting aside validation issues is compatible with the idea that the argument view offers insight into the general epistemological functioning of CSs. We view this response with some skepticism and suggest that the often “motley” (Winsberg 2019) nature of validation raises doubts about whether the epistemic power of CSs can be captured by focusing on the way in which simulation outputs follow from given inputs.

Second, we outline that when using CS as a means to acquire new knowledge one often also draws on practices of inductive reasoning whose role does *not* consist in demonstrating the truth of the premises of the reconstructing argument. To the extent that the argument view falls short of properly incorporating these practices, it fails to meet its own demand of satisfactorily answering (Q_CS).

5.1 Validation

Remember that for Beisbart, the statement that a computer simulation has the same epistemic power as an argument implies that computer simulations and arguments contribute to the acquisition of new knowledge in a similar way. So far, we have agreed with Beisbart that the relationship between simulation input and output is comparable to the relationship between the premises and conclusions of deductive arguments. Yet, if our interest concerns the general way in which CSs contribute to acquiring new knowledge, do we not need to take into account more than just the way in which a single simulation run generates its output?

It seems clear that validation, broadly understood as comprising “the efforts to show that a computer simulation represents its target appropriately” (Beisbart 2019, 36), plays a crucial role in assessing the way in which CSs contribute to knowledge acquisition. Beisbart himself admits that by abstracting from validatory activities, the argument view does not answer all of the relevant epistemological questions about CSs:

> Our reconstructions abstract from the problems of validation. They do not explain how the working scientists come to think that the algorithm used in the simulations reflects the dynamics of the target […]. In this sense, our reconstructions abstract from important epistemological problems about computer simulations. (Beisbart 2012, 417)

Nevertheless, Beisbart proposes that the additional considerations needed to draw a connection between CSs and their targets also take an *inferential form*. As he puts it, “validation is typically inferential too” (417). Furthermore, he stresses that while the “argument that reconstructs the one run of the simulations will be deductive, […] the further inferences run by the simulationalist are likely to be inductive” (413).

Indeed, one would certainly (and almost trivially) grant that in justifying that the simulation model is an adequate means of learning about a physical target, one crucially relies on drawing additional *inferences* beyond those that lead to the simulation results as their conclusion. However, consider that in the case of experiments, one would likewise say that validatory arguments are usually required when claiming that experimental results help us learn about the respective subject of scientific investigation. While there is a debate as to whether the arguments given as legitimization for the inference from result to target in CSs differ in nature from those given for experiments (cf. Morrison 2009; Winsberg 2009a), it seems clear *that* arguments are central when claiming that knowledge about a physical target has been generated. If at the same time, however, a central idea underlying the argument view is to understand their epistemic functioning precisely by setting them *apart* from experiments, then pointing to the fact that validation is inferential too confuses the issue rather than clarifying it.[39]

Still, Beisbart proceeds by providing an additional reason that setting aside validation does not threaten the argument view. He argues that despite the need for further validatory work to be done in order to "transfer knowledge from the model to the target" (Beisbart 2012, 419), comparing CSs to arguments informs us about the general epistemic *functioning* of CSs.

Considering that he explicitly states that performing validatory activities means verifying whether the premises of a CS are true, it seems that to him, admitting the need for epistemic activities that *go beyond* drawing deductive inferences generally fits in the CS-as-arguments picture rather than contradicting it:

> Simulation scientists have […] to show that their simulations faithfully reflect the target in the intended respects. In the framework developed in this paper, this is to show that the premises of the reconstructing argument are true. […] This part is called *validation* of the simulations. Validation is supposed to make a case for the results of a computer simulation study in relation to its target. (Beisbart 2012, 417, emphasis in original)

Let us put these pieces together: Beisbart explicitly grants that additional non-deductive practices usually play a role when referring to CS as means of knowledge acquisition. At the same time, he suggests that these additional practices serve to evaluate the truth of the

[39] Interestingly enough, it has indeed been suggested that *experiments* can be characterized as arguments. As Hon (2000) argues, experiments can be cast as arguments because the conclusions of an experiment "may be seen as the result of a chain of reasoning concerning the behavior of some material system." While we cannot, in the scope of this paper, further elaborate on Hon's proposal, it seems clear that characterizing validation as inferential does help to make a distinction between CSs and experiments.

premises of the reconstructing argument. To summarize, the key idea seems to be that if all epistemic activities relevant to the capacity of CSs to generate new knowledge concern the process of either drawing inferences or showing that the premises of a CS are true, then one can maintain the idea that, overall, the epistemic power of the CS should be that of the respective reconstructing argument.

We can make sense of CSs' contribution to knowledge acquisition precisely by acknowledging that they enable new insights by means of 'inferentially reliable' processes rather than through interaction with the world. The idea seems to be that the argument view provides an answer to (Q_CS) *indirectly* because it renders intelligible how CSs help us gain new knowledge about a model, which is *then* transferred to a target. More precisely, as indicated above, Beisbart suggests that within the framework of the argument view, performing validation activities is equivalent to verifying that the premises of an argument are true. It is acceptable to abstract from validation because validatory activities play a "confirmatory" role with regard to the inferences drawn: by promoting an argument view on CS one emphasizes the fact that knowledge is being generated inferentially, and that to validate a CS is to show that the premises of the respective inference are true.

Does this claim hold? Consider that in the philosophical literature there is a debate on whether the verification and validation (V&V) of CSs can be separated. For example, Lenhard (2019, 208) argues that "it is not possible to first verify that a simulation model is 'right' before starting to tackle the 'external' question of whether it is the right model." Consider, for example, the case of parameter assignments. On the one hand, according to Lenhard, the adjustment of parameters is part of an improvement of the *performance* of a discretized computational model, i.e., the part where one makes sure that it appropriately 'fits' the theoretical model. At the same time, however, the assignment of parameters also concerns the way in which the model is being *characterized* and thus also concerns validatory questions. As Lenhard puts it: "without the assignment of parameters, neither the question about representational adequacy nor the question about behavioral fit can be addressed" (207).

In a similar direction, Winsberg (2009b, 839) states that in cases in which one finds that the computational model does not account for experimental data, it may be unclear whether one should "blame the underlying model or […] the modeling assumptions used to transform the underlying model into a computationally tractable algorithm." However, in contrast to

Winsberg's and Lenhard's positions, Margaret Morrison (2015, 285ff.) holds that by stating that V&V are inherently entangled, one mistakenly suggests that performing V&V is a "piecemeal activity" when in fact it is important for the credibility of CSs to stress that these activities are usually undertaken in a sequential manner[40] and can be clearly distinguished conceptually as well as methodologically.

Yet, whether or not one agrees with Morrison that Winsberg and Lenhard overestimate the entanglement of V&V, it is remarkable that she herself points out that one should not simply think of validation as mere 'hypothesis testing': "the validation process involves providing evidence for the accuracy of a particular model rather than establishing a 'true/false' conclusion" (Morrison 2015, 282). Rather, as she observes, a central challenge lies in the fact that different sources of uncertainty are involved and that the goal of V&V methodology is to manage uncertainties that arise, for example, due to parameter choice or a loss of information in the process of discretization (250ff.). Moreover, she points out that in many cases, validation experiments "have poorly characterised or very little data input for computational simulations" which may be associated with the need for a "probabilistic treatment of uncertain parameters in the computational submodels, initial conditions, or boundary conditions" (278). As she puts it, the question whether a validation experiment legitimately confirms that a computational model is accurate often depends crucially on how 'accuracy' is defined (Morrison 2014, 945). This, in turn, seems far from an easy and straightforward task. Related to such concerns, Winsberg (2019) emphasizes that "most model equations chosen for simulation are not in any straightforward sense 'the right equations'" as they may result from compromises between what is considered an adequate description of the phenomenon on the one hand and computational tractability on the other.[41]

Against this backdrop, the assumption that validation consists in checking whether the premises of an argument are 'true' seems to be a simplistic description of the challenges that are encountered in the process of validating a CS. Far from being straightforward or 'linear', modeling in the context of CSs is a complex process that presents the modeler with varying

[40] Morrison argues that if validation were performed before verification, one would run the risk of arriving at an untenable result and being unable to assess where the problem lies (cf. Morrison 2015, 265).

[41] For example, the theoretical model with which the modeling process usually begins in the context of CSs often has the form of some differential equations, which describe the dynamic behavior of the target system. These equations usually will not have analytic solutions, which comes with the challenge of relying on modeling decisions that tend to be "theoretically unprincipled", but at the same time is a central reason for the usefulness of CS (Winsberg 2010, 7ff.)

degrees of freedom in their modeling choices at different points in the modeling process as well as with the challenge of dealing with a variety of uncertainties. While we may well abstract from challenges arising in the context of validation if the aim is solely to capture certain aspects of CSs, it is doubtful whether we can do so when trying to answer (Q_CS).

5.2 CSs and the inductive leap

Thus far, we have argued that although the argument view addresses the need for additional validatory activities in order to formulate knowledge claims on the basis of the results of a CS, it does not adequately capture the epistemic importance of these activities. In the following discussion, we offer an additional reason for rejecting the idea that the argument view provides a coherent picture of how CSs contribute to knowledge acquisition. We do so by demonstrating that there are often other inferential practices involved in dealing with CS that are crucial for assessing the role of CSs in contexts of knowledge acquisition. These practices of reasoning are, as we will show, not appropriately addressed by the argument view as they are neither limited to a validatory role nor fully captured by what can be inferred deductively from the input of single simulation runs[42].

Let us begin with a rather abstract example. In his 1997 paper, Mark Bedau argues that some CSs grant epistemic access to weakly emergent phenomena. According to Bedau (1997, 377f.), a macrostate *P* of a system *S* with microdynamic *D* is weakly emergent iff *P* can be derived from *D* and *S*'s external conditions but only by means of simulation. In order to illustrate the concept of weak emergence, Bedau discusses the case of Conway's Game of Life, a cellular automaton that consists of a grid of cells whose states change depending on the states of their neighboring cells over a number of discrete time steps. When simulating the evolution of Life configurations over time, one can observe the emergence of interesting macropatterns such as gliders.[43]

As Bedau argues, whether patterns such as gliders will emerge in instances of Life is fully dependent on (a) the initial state of the system (in this case, the microstates of the cells of a

[42] Beisbart (2012) did of course not forget about these practices. In his paper he focuses "on one single run of a simulation and not on a CS study" (424) and thereby excludes the inductive reasoning practices we are discussing in this section. We believe, however, that it is an important part of the answer to (Q_CS).

[43] Gliders are patterns (consisting of 'live' cells) that travel diagonally across the Life grid.

Life instance after zero time steps) and (b) the microdynamics (in this case, the rules according to which the states of the respective cells change). However, he points out that having complete information about the microrules and initial configuration does not suffice for an epistemic agent to estimate whether random initial Life configurations will spawn certain macropatterns such as gliders.

Even though it seems that *in principle* the occurrence of patterns could be predicted on the basis of such information, Bedau observes that *in practice* studying the evolution of Life configurations over time by means of computer simulation is the only way for epistemic agents to determine whether a certain microconfiguration will lead to the occurrence of weakly emergent phenomena.[44]

In the case of Life, we may be interested in knowing the likelihood that a given random initial configuration will spawn gliders after a number of iterations. We may thus seek a law that specifies the probability of an initial configuration quickly leading to a macrostate in which gliders appear. As Bedau (1997, 385) states, the fact that patterns such as gliders are weakly emergent "does not prevent us from readily discovering various laws" about them. By means of "extensive enough observation", one can see, for example, that for most configurations gliders will spawn quickly, and summarize this information in the form of a probabilistic law. However, as he puts it, "[empirical] observation is generally the only way to discover these laws" (385).

Thus, in the case of Life, computer simulations play not only a heuristic role in illustrating the development of Life configurations over time but also an epistemically *irreducible* role in identifying laws according to which the emergence of certain patterns can be quantified.

Even though we are not primarily interested in the conception of "weak emergence"[45], Bedau's considerations are particularly interesting for two reasons:

[44] There is a debate about whether weakly emergent phenomena in fact qualify as *emergent*. We do not want to delve deeper into this discussion here. As we are interested in the question of how scientists as epistemic agents can learn from CSs, it suffices to say that it does not seem possible for them to proceed in any way other than an inductive manner when seeking to learn about the emergence of certain patterns or phenomena.

[45] We do not mean to imply that CSs' contribution to knowledge generation goes beyond what can be deductively inferred from their 'premises' only in cases in which one is interested in phenomena that qualify as 'weakly emergent'. Rather, we suggest that Bedau's argumentation indicates particularly clearly that, more generally, scientists may often be interested in *general* rules of pattern emergence rather than in a single simulation output and that in such cases inductive inferences are central *despite* outputs being derived deductively.

(I) Given that weakly emergent structures or macropatterns can in principle be derived from a complete knowledge of the initial conditions and algorithmic rules of a system, CSs that serve as a means of tracing the behavior of such structures can be described in full accordance with the reconstruction thesis.

(II) Following Bedau, CSs can serve as a means of acquiring knowledge about rules that describe the occurrence or behavior of such weakly emergent phenomena. These rules can, however, be identified *by no other means than empirical observation* of the CSs' behavior.

The second point raises questions that are relevant to the discussion of similarities between CSs and arguments: If we want to maintain that the epistemic power of CSs is the same as that of deductive arguments, then how can we make sense of the fact that learning about interesting characteristics of Life depends on *empirical observation* of the simulation? Moreover, if formulating empirical laws about the behavior of Life requires studying *numerous* simulation runs with *different* initial configurations, does the fact that single runs of a simulation can be reconstructed as arguments satisfactorily illuminate how they contribute to knowledge acquisition?

An obvious objection to these concerns is that studying the emergence of Life patterns does not really address the acquisition of knowledge about – in Beisbart's words – 'real-world phenomena'. Yet there are interesting parallels to other cases in which one attempts to generate 'model-based knowledge' about the world.

Discussing the role and characteristics of theoretical models in economics, Sugden (2000; 2009) criticizes the idea that the central mode of knowledge acquisition about real-world targets is to deductively draw conclusions from well-founded assumptions. Sugden considers cases such as Akervlov's market for lemons or Schelling's checkerboard model of residential segregation. The latter is an agent-based model consisting of a grid with cells 'inhabited' by agents that are members of one of two groups and are ascribed a certain preference for living beside agents of the same (e.g. ethnic or economic) group. Over the course of time, agents 'move' across the board according to their preferences. As Sugden argues, "we have to realize that results that [Akerlov and Schelling] derive deductively within their models are not the same as the hypotheses that they want us to entertain" (2000, 18). Even though concrete 'outputs' are generated by means of deductive inference, he suggests that usually it is *not these outputs themselves* that are the sole subject of interest. Rather, he claims that one is often interested in drawing more *general* statements about the world that are inferred *inductively*. In the case of the checkerboard model, for example, Sugden (17) argues as follows: Rather than starting with a set of well-grounded premises and ending with deductively inferred conclusions that follow from these premises, it seems that one is often interested in reaching *more general* conclusions and thereby relies on drawing explanatory, predictive or abductive links. If, for example, in real-world contexts, one can observe some phenomenon *X* and if *X* is – within the scope of a model – induced by *Y*, then one might want to generally infer that *Y may* be the central cause for the occurrence of *X*. In this case, however, an 'inductive leap' is present. What is relevant for Sugden is that in either case, despite model results being inferred deductively, certain (relevant) statements about real-world targets seem to depend on further inferences that are inductive. Taking the example of the checkerboard model, he suggests:

> [H]aving experimented with Schelling's checkerboard model with various parameter values, I have found that the regularity described by Schelling persistently occurs. Having read Schelling and having thought about these results, I think I have some feel for why this regularity occurs; but I cannot give any proof that it must occur (or even that it must occur with high probability). My confidence that I would find similar results were I to use different parameter values is an inductive inference. (22f.)

Based on Sugden's considerations, let us consider two important ways in which the checkerboard model may be regarded as a (potential) means of providing new knowledge about the world: Firstly, given *that* segregation is observed in real-world cities, the checkerboard model might indicate that individuals' preferences may be one relevant driving force for this phenomenon. Secondly, by studying the dynamics of *several* model instances with different parameters, one might infer that Schelling's model provides insights on the surprising *rapidity* with which segregation occurs even if people's preferences for living next to members of their own group are rather mild (see Aydinonat 2007, 437f.). In both cases, knowledge claims would rely centrally on drawing inductive inferences and, especially in the second case, on making a *general* statement about segregation dynamics based on an observation of the outcomes of *different* model 'runs'.

One might still worry that concentrating on theoretical models in economics sets a too narrow focus and that the situation is different for most scientific practices in which CSs are used to learn about real-world phenomena. Consider, however, a final example that focusses on some aspects of the use of CSs in climate science and meteorology.

Climate science was one of the first fields to use computer simulations on a large scale. Already in the 1950s a computer model existed, which was used to simulate the dynamics of the atmosphere. Wendy Parker (2014: 337) describes that the scientists who used these simulations were interested in weather forecasts and the development of the climate as well as advancing their theoretical understanding of the climate system. One main impact of the introduction of the computer in climate science was that it enabled the scientists to get approximate solutions for the theoretical equations they assumed to govern the climate system, which would have taken too much time to calculate by hand. The computer "helped scientists to see what followed (or failed to follow) from the physical assumptions reflected in different mathematical models of the atmosphere or climate system" (341). According to Jule Charney (1955), one of the meteorologists who developed these early models, the "radical alteration" that was happening in meteorology with the introduction of CSs was grounded in "its ability to serve as an inductive device." (798 f.) Controlled meteorology experiments are difficult or impossible, but the CSs allowed for hypothesis testing and thereby permitted "a wider range of inductive methods". (799)

As Parker (2014: 348 ff.) has discussed, climate models have for example been used to test the hypothesis that human activities are causally connected to the global warming in the second half of the twentieth century. In order to test this hypothesis, one could first use a climate model with the dynamics of the natural forcing factors but with fixed anthropogenic forcing factors and observe how global warming develops in this simulation. In a second simulation, one adds the anthropogenic forcing factors. If global warming could be observed in the second simulation but not in the first one, this would be evidence for the hypothesis.

The above example fits the structure of eliminative induction (Pietsch 2004). Therefore, if we conclude from the results of the proposed simulations that human activities are causally connected to global warming, this conclusion is an inductive one.

Given the aforementioned examples, it seems that CSs are commonly used to make statements about real-world targets that *go beyond* what is deductively derived from initial assumptions (and thereby that it is common to rely on practices of inductive reasoning).

If we were to argue that simulations contribute to knowledge acquisition specifically by deductively inferring an output from some initial conditions and transformation rules, would we not be missing an important aspect of their contribution to knowledge acquisition? To the extent that simulations are considered a means of drawing more general conclusions – for example, about the key forces that cause some macroscopic pattern to occur or about the relationship between different characteristics of a physical target – and to the extent that the argument view fails to take these practices of reasoning into account, it cannot satisfactorily answer (Q_CS).

6 Conclusion

In this paper, we have critically discussed Beisbart's argument view on CS. The argument view consists of three different claims: the reconstruction thesis, the epistemic power thesis and the execution thesis. Based on these three theses, Beisbart seeks to provide an answer to the question of how scientists can gain new knowledge when using a CS.

We grant that, as Beisbart suggests, CSs can be reconstructed as arguments whose premises are the initial state and the transformation rules upon which the respective simulation is based. However, we disagree with the epistemic power thesis. Beisbart claims that CSs have the same epistemic power as arguments because they can contribute to the acquisition of new knowledge in the same way as arguments. This analogy falters in an essential respect: while one might grant that the relationship between simulation input and output can be understood as a relationship between premises and deductively derived conclusion, the character of this relationship alone does not provide a satisfactory account of how CSs contribute to the acquisition of new knowledge.

In order to substantiate this claim, we have illustrated that clarifying the role of CS in knowledge acquisition requires properly taking account of the central involvement of practices of reasoning that are genuinely *inductive*. We have pointed to different cases in which the role of CSs in helping epistemic agents to gain new knowledge can only be satisfactorily understood if one takes into account practices of inductive reasoning that go beyond what is deductively inferred in the course of a single run of a CS. We therefore conclude that the reconstruction argument does not fully capture the epistemic power of CS. In addition, the argument view largely disregards the epistemological problems concerning the validation of CS, which, however, seem to be central to a proper grasp of the contributions (and challenges) of CSs in contexts of knowledge acquisition. This casts further doubt on the capacity of the argument view to answer (Q_CS).

Nevertheless, Beisbart's suggestion of *reconstructing* the simulation run as an argument does have its benefits, as it indicates that the central challenge of epistemic opacity might be less about the necessity of checking every computational step of a simulation than it is about the importance of facilitating strategies to manage and address sources of uncertainty and error that are present in the modeling process.

References

Appel, Kenneth I. and Wolfgang Haken. 1989. “Every planar map is four colorable″ (Vol. 98). Series: Contemporary mathematics. American Mathematical Soc.

Aydinonat, N. Emrah. 2007. “Models, conjectures and exploration: an analysis of Schelling’s checkerboard model of residential segregation”. Journal of Economic Methodology 14(4): 429–54.

Bedau, Mark A. 1997. “Weak Emergence”. Noûs 31(s11): 375–99.

Beisbart, Claus. 2012. “How Can Computer Simulations Produce New Knowledge?” European Journal for Philosophy of Science 2(3): 395–434.

Beisbart, Claus. 2019. “What Is Validation of Computer Simulations? Toward a Clarification of the Concept of Validation and of Related Notions”. In: Computer Simulation Validation, Claus Beisbart and Nicole J. Saam (eds.). Cham: Springer International Publishing, 35–67.

Beisbart, Claus, and John D. Norton. 2012. “Why Monte Carlo Simulations Are Inferences and Not Experiments“. International Studies in the Philosophy of Science 26(4): 403–22.

Boge, Florian J. “Why computer simulations are not inferences, and in what sense they are experiments." European Journal for Philosophy of Science 9.1 (2019): 1-30.

Charney, Jule G. 1955. “Numerical Methods in Dynamical Meteorology.” Proceedings of the National Academy of Science 41(11): 798–802.

Hon, Giora. 2000. “The Limits of Experimental Method. Experimenting on an Entangled System: The Case of Biology”. In Science at century’s end: Philosophical questions on the progress and limits of science, Martin Carrier, Gerald J. Massey, and Laura Ruetsche (eds.). University of Pittsburgh Press, 284–307.

Humphreys, Paul. 2004. Extending Ourselves: Computational Science, Empiricism, and Scientific Method. New York: Oxford University Press.

Humphreys, Paul. 2009. “The Philosophical Novelty of Computer Simulation Methods“. Synthese 169(3): 615–26.

Kaminski, Andreas. 2018. „Mathematische Opazität. Über Rechtfertigung und Reproduzierbarkeit in der Computersimulation“. In: Arbeit und Spiel. Alexander Friedrich, Petra Gehring and Christoph Hubig (eds.). Nomos, 253–78.

Kroll, Joshua A. 2018. “The fallacy of inscrutability.” Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences 376.2133: 20180084.

Kuorikoski, Jaakko. 2012. “Simulation and the Sense of Understanding”. In: Models, Simulations and Representations. Paul Humphreys and Cyrille Imbert (eds.). New York, London: Routledge, 168–87.

Kuorikoski, Jaakko, and Aki Lehtinen. 2009. “Incredible Worlds, Credible Results”. Erkenntnis 70(1): 119–31.

Lenhard, Johannes. 2006. “Surprised by a Nanowire: Simulation, Control, and Understanding”. Philosophy of Science 73(5): 605–16.

Lenhard, Johannes. 2019. Calculated Surprises: A Philosophy of Computer Simulation. New York: Oxford University Press.

Massimi, Michela, and Wahid Bhimji. 2015. "Computer Simulations and Experiments: The Case of the Higgs Boson". Studies in History and Philosophy of Science Part B: Studies in History and Philosophy of Modern Physics 51: 71–81.

Morgan, Mary S. 2005. "Experiments versus Models: New Phenomena, Inference and Surprise". Journal of Economic Methodology 12(2): 317–329.

Morrison, Margaret. 2009. "Models, Measurement and Computer Simulation: The Changing Face of Experimentation". Philosophical Studies 143(1): 33–57.

Morrison, Margaret. 2014. "Values and Uncertainty in Simulation Models". *Erkenntnis* 79(S5): 939–59.

Morrison, Margaret. 2015. Reconstructing Reality: Models, Mathematics, and Simulations. New York: Oxford University Press.

Newman, Julian. 2015. "Epistemic opacity, confirmation holism and technical debt: Computer simulation in the light of empirical software engineering." International Conference on the History and Philosophy of Computing. Springer, Cham.

Norton, John D. 2004. "On Thought Experiments: Is There More to the Argument?" Philosophy of Science 71(5): 1139–51.

Parker, Wendy S. 2009. "Does matter really matter? Computer simulations, experiments, and materiality". Synthese 169(3): 483–496.

Parker, Wendy S. "Simulation and understanding in the study of weather and climate." Perspectives on Science 22.3 (2014): 336-356.

Pietsch, Wolfgang. "The structure of causal evidence based on eliminative induction." *Topoi* 33, no. 2 (2014): 421-435.

Stöckler, Manfred. 2000. "On modeling and simulations as instruments for the study of complex systems". In: Science at century's end: Philosophical questions on the progress and limits of science. Martin Carrier, Gerald J. Massey, and Laura Ruetsche (eds.). University of Pittsburgh Press, 355–73.

Sugden, Robert. 2000. "Credible Worlds: The Status of Theoretical Models in Economics". Journal of Economic Methodology 7(1): 1–31.

Sugden, Robert. 2009. "Credible Worlds, Capacities and Mechanisms". Erkenntnis 70(1): 3–27.

Tymoczko, Thomas. 1979. "The Four-Color Problem and Its Philosophical Significance". The Journal of Philosophy 76(2): 57.

Winsberg, Eric. 1999. "Sanctioning Models: The Epistemology of Simulation". Science in Context 12(2): 275–92.

Winsberg, Eric. 2009a. "A tale of two methods". Synthese 169(3): 575–592.

Winsberg, Eric. 2009b. "Computer Simulation and the Philosophy of Science". Philosophy Compass 4(5): 835–45.

Winsberg, Eric. 2010. Science in the Age of Computer Simulation. Chicago: The University of Chicago Press.

Winsberg, Eric. 2019. "Computer Simulations in Science". In The Stanford Encyclopedia of Philosophy, Edward N. Zalta (eds.). Metaphysics Research Lab, Stanford University. https://plato.stanford.edu/archives/win2019/entries/simulations-science/ (12. November 2020).

Chapter 4: Model building in theoretical ecology: Is ‘species’ a good observable?
Or: Why grass is not less important than rabbits

Abstract

The main goal of theoretical ecology is to understand and explain phenomena of the living world. One central epistemic tool for this purpose are models. In this paper, we discuss two difficulties of the relations between models, theory and observation. The first challenge, shared among all data-based models, is the difference between the phenomenon and the data of the phenomenon that we use to build the model. Secondly, we address a problem that is more specific to ecological models, namely the choice of ‘species’ as the central observable in ecology. Organisms, which are easily observable, are described in greater detail and with clearer distinction than less easily observed organisms, leading to a skewed representation in the data. Nevertheless, ecological models and theories have shown to work well across a range of gradients and scales. To understand this seeming contradiction better, we address general problems of model building using food web ecology as an example case.

1 Introduction

The main goal of theoretical ecology is to understand and explain organismal diversity and interactions among organisms and their environment (Begon and Townsend 2021). By combining results from descriptive observations of natural ecosystems and empirical studies both inside and outside of the lab, theoretical ecologists detect patterns in natural complexity and attempt to build models based on biologically realistic assumptions, which reproduce these patterns. These idealised model-representations of ecological systems aim to improve the understanding of the natural world by uncovering fundamental mechanisms and biological processes.

The process of gathering knowledge in theoretical ecology is based on two main assumptions. The first assumption is the existence of fundamental processes in ecology. Only if there are general underlying mechanisms generating natural complexity across species and ecological environments, theoretical ecology can ever be able to detect any of these in observable phenomena. Fundamental research has a long tradition in the sciences in general as well as in ecology (Scheiner and Willig 2008; Marquet et al. 2014) and has contributed to deepened understanding of natural phenomena, has led to economic, social and political progress and has served as a source for innovation (Courchamp et al. 2015). Hence, this first assumption is widely acknowledged as acceptable and useful across scientific disciplines.

The second assumption is shared among all observation- or data-driven sciences. Models in data-driven sciences are built based on data of phenomena that has been gathered via *observations*. The assumption is that one can gain knowledge about the phenomenon from this data by using inductive and possibly abductive reasoning (Mazzocchi 2015). Thus the success of a model as a tool for understanding natural phenomena relies on the relation between the phenomena and the observations of the phenomena. The aim of this paper is to discuss this second assumption in the scope of ecology using insights from the philosophy of science discussion about model building. We believe that this contribution to the philosophy of ecology can benefit both the philosophy of science by adding an example with new facets to the discussion about the role of models in science and the field of theoretical ecology by reflecting on the assumptions and idealizations of model building in ecology.

In the next section, we describe the problems of incomplete data sets and the role of species as an observable in theoretical ecology. In section 3, we introduce the model building process from a perspective of philosophy of science before we combine the two parts in section 4 using the example of food web ecology.

2 Data and Observables in Theoretical Ecology

Ecological theory building is heavily dependent on the quality of the underlying data. Ecosystems are highly complex systems with a vast number of agents and interactions, feedbacks and nonlinear responses. Biological organism sizes, habitat areas and temporal scales span a huge range. Furthermore, ecological systems are open systems with regards to energy and nutrient fluxes as well as indirect interaction effects. All these key characteristics of ecological phenomena make it close to impossible to gather exhaustive observational data of ecosystems. On top of the fact that ecological data sets are generally incomplete, they are very dependent on the chosen system boundaries. All of these factors heavily complicate theory building and theory unification across scales (Turner et al. 1989; Proulx 2007; Marquet et al. 2014).

While the problem of incomplete data sets for the relation between phenomena, data and models applies to most empirical sciences, ecological data sets have an additional complication: the choice of 'species' as the central observable in ecology. A species is typically defined genetically or by the capability to reproduce with fertile offspring. Naturally, species, which are easily observable, are described in greater detail and with clearer distinction than less easily observed organisms. Human perception makes it easier to distinguish the common rabbit (*Oryctolagus cuniculus)* from the European hare (*Lepus europaeus*) than the perennial ryegrass (*Lolium perenne*) from the Kentucky bluegrass (*Poa pratensis*) - two common grass species found on middle European lawns. As a consequence, ecological data sets are not only incomplete but also highly skewed. When looking for example at food webs[46] and predicting the ratio of predators and prey species, the bias for larger species - which happen to often be higher up the food web than smaller species - systematically affects the empirical distribution and hence the derived models and theories (compare Figure 4).

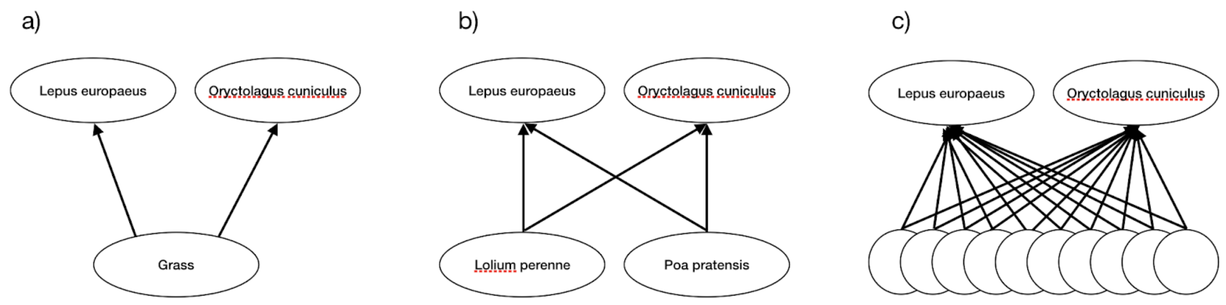

Figure 4: Three representations of a food web with one rabbit and one hare species, feeding on grass. The simplest representation is to lump all grass species into one data point (a). If you know, that the two most common grass species in European lawns are present, you could decide to treat them separately (b). In reality however, lawns and meadows are usually very diverse with regards to their species composition (c), even though the different grass species look very similar at first sight. If one wants to analyse the ratio of primary producer to consumer species in the current examples, one arrives at ranges from ½ (a) to 5 (c).

[46] For a definition of food webs, see for example Hui (2012, 6): "Food web is an important conceptual tool for illustrating the feeding relationships among species within a community, revealing species interactions and community structure, and understanding the dynamics of energy transfer in an ecosystem."

The two problems – incomplete data sets and a systematic resolution bias for larger, more easily observable species – pose fundamental challenges to the formation of ecological theories. Nevertheless, theoretical ecology has developed a large diversity of models and theories, which provide good predictions both for specific systems and across systems and scales, which have tremendously advanced the field and natural sciences in general (Hastings and Gross 2012). Interestingly, ecologists even have theories, which predict the degree distributions in food webs accurately (Williams 2010).

This seems to be surprising given the described problems with the data. Explaining how ecologist are able to describe ecosystems well and make good predictions about them with these skewed models would not only be valuable for ecologists but is also of interest to philosophers of science. Of course, this is not the only case in which scientists aim to learn something about phenomena with models, which contain misrepresentations. Philosophers of science give different explanations why this can be a fruitful undertaking (see for example: Cartwright 1983; Morgan 1999; Frigg and Hartmann 2020). Let us look at some of the cases:

It is quite common for models to contain artificial parameters, which do not represent a specific part of the phenomenon. One such example is the Arakawa operator, which was introduced by Akio Arakawa (1966) to fix issues with the stability of the dynamics of one of the first ever simulations of the circulation of the earth’s atmosphere. The assumptions Arakawa had to add to the model to introduce his operator were neither compatible with physical theory nor founded in experimental experience but rather went strictly against them. They were only accepted because of the success of the resulting model (Lenhard 2007). This is different from the case of species in ecological models however since no artificial component has been introduced. Even though some of the parameters are simplified, all of them refer to actual properties of the system.

Another interesting comparison are early meteorologist models and Lorenz’ discovery that humans cannot measure the relevant input data (for example temperature or humidity) exact enough to achieve successful long-term predictions. Even arbitrarily small differences in parameters can influence the entire system significantly (later phrased as the ‘butterfly effect’). This discovery contributed significantly to the development of chaos theory (Lorenz 1963). The measurement problems that ecologist have with species of different sizes do however not fall in the same category. There is no principle reason why species cannot be distinguished with equal precision on different scales.

There has also been some discussion in philosophy of ecology on the matter. Colyvan et al. (2009) offer two possible answers why simplified and idealized models might still give us useful information about the phenomenon. Using the example of assuming a constant carrying capacity in the model for a system with a varying carrying capacity, they argue that the modeler might – with experience and through trial and error – find simplifications, which do not impact the model predictions significantly and thereby generating a useful model. They also offer a more positive answer: Simplifications might be useful abstractions from irrelevant details to reveal the core mechanisms of the system. Again, none of these descriptions fit well for the case of species as it does not seem to be the case that the modelers actively decide to misrepresent smaller species in the food webs but are rather forced to do so because of the available data. All in all, it seems that the example of species in ecological models might be a useful addition to the philosophical debate about models and should be discussed in some more detail.

To understand the underlying problems more rigorously and to analyse potential solutions and future implications for theoretical ecology, we will first look at model formation from the perspective of the philosophy of science and epistemology and then discuss the application to theoretical ecology using food web theory as an example.

3 Models in philosophy of science

Models have become a central epistemic tool in many scientific disciplines. One of their functions is to represent phenomena. The goal is to learn about features of the target system[47] by studying and experimenting with the model. Whether knowledge that has been acquired about the model can be transferred onto the target system at all is the source of ongoing debates in philosophy of science (Giere 2004; Bailer-Jones 2009; Frigg and Nguyen 2017). A central aspect to this debate is the question how models relate to the target system, i.e. how they *represent*. This relation has to be understood before the question of a possible

[47] Target or Target system has become the established notion to refer to the real-world phenomenon or system that is investigated with scientific methods and represented by the model.

knowledge transfer from model to target system can be addressed properly. The model building process is fundamental to understanding the relation of a model to its target system. In the following we discuss the model building process and the challenges one has to face especially in data-driven sciences to get from the target system to a model of the target system.

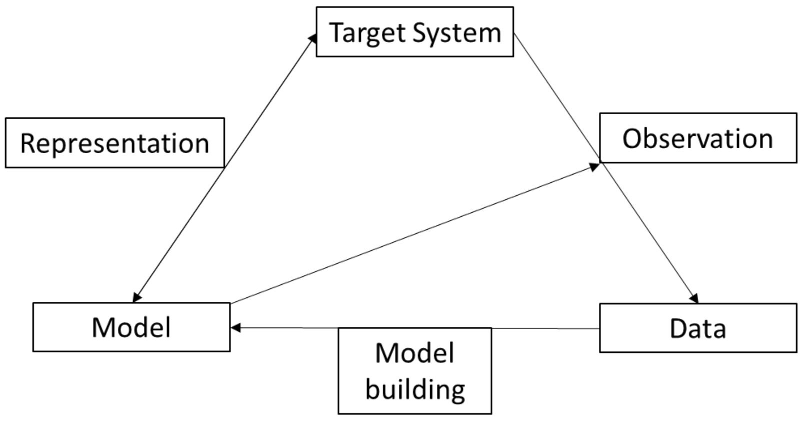

Figure 5: Central elements of the model building process (target system, data, model) and their relationships (observation, model building, representation). The arrows describe the dependencies between the elements.

Figure 5 displays the relationships of central elements of the model building process. The target system is a phenomenon the scientist is interested in. Data is the empirical information that has been gathered about the target system via observation. A model is a representation of the target system, built from theoretical considerations as well as empirical data and validated against the existing data.

All of these elements as well as the relationships between them bring about epistemic challenges. Without claiming to be exhaustive, we will touch upon a selection of these challenges. We will discuss them theoretically at first and give examples of these problems from the praxis of theoretical ecology in the next chapter.

3.1 Observation

The target system is a selected phenomenon that we choose to investigate. It can be nearly everything that can be of scientific interested, e.g. the neural network of the human brain, the melting point of a specific metal or the dynamics of populations in a predator-prey interaction. We use observation and experiments to gather data about instances of these phenomena and try to find patterns in the data to learn something about the phenomena. In the case of the melting point of a metal for example, the scientist will perform a number of experiments, heating the metal and measuring the temperature at which the metal starts melting. The results of the measurements will most likely differ by a small margin, but using the mean of the collected data-points, the scientist will get a good estimate of the phenomenon he tries to learn something about – the “true” melting point of the metal. It is important to note the difference between the phenomenon itself, an assumed fundamental process in reality, and the data of the phenomenon, the results of the observation that include known and unknown errors (Bogen and Woodward 1988).

If we want to learn about the target system via data, we are dependent on experiment and observation. This dependence brings with it a new set of epistemic challenges that have been discussed at length in philosophy of science[48]. The data that is acquired via observation will always be incomplete, because it can never be guaranteed that all instances of the phenomenon have been observed, all potential constellations of the chosen variables have occurred or even that all the relevant factors have been identified and are observable in the chosen target system. Furthermore, the data will always be biased by the choice of observables[49], because this choice leads to a categorization of the data that affects the structure and the description of the data. In this way the choices how the models are built have influence on future observations (see example in the next chapter).

3.2 Model building

The construction of the model is based on theories, which describe the phenomenon under investigation and on data, which has been gathered about the target system. The impact of theories on the model building process depends on the status of the theories in terms of development and reliability in relation to the phenomenon under investigation. If the theory is well-developed and extensively tested, the model will typically reflect the equations of the

[48] For an overview see: Bogen 2017.
[49] Observables in this context describes the variables that are used to describe the state of a system. Examples for observables in physics are momentum and position, for ecology it would for example be species.

theory[50]. In this case, the main purpose of the model is to test the hypotheses and predictions, which follow from the theory against the data. In case there are no detailed theoretical descriptions of the specific phenomenon or the existing theory cannot be modelled, data is the driving force in the model building process. The model will include general theoretical principles and assumptions about structural features of the phenomenon. The main input however will be from patterns and regularities in the observed data and the fitting of the model to the existing data. The second case is more commonly found in ecology, as ecological theories are mainly understood as patterns or regularities across models. In both cases, the model is partially independent from both theory and data as it is neither defined purely through the theoretical equations nor is it a reconstruction of the data alone. Models have therefore been phrased as mediators between theory and experiment (Morgan and Morrison 1999).

There is no algorithm, which defines how to construct a model given a set of theoretical sentences and a set of data, and there is more than one way to construct a consistent model that fits a specific set of data (problem of underdetermination). The modeler has to choose the equations that shall represent the phenomenon and has to decide which parameters in the model are fixed and which are variable and can be adjusted to fit the data. This means that in particular the choice of observables is not given by the target system. Different sets of observables are possible and will have different strengths and weaknesses.

3.3 Representation

Once the model has been completed, one has to make sure, whether assumptions about the represented phenomenon are correctly implemented (verification) and whether the model correctly describes the gathered data (validation). Both of these processes can be far from trivial[51].

[50] This does not imply that the process of model building is trivial in this case. For each case the modeler has to choose which parts of the theory are relevant for the specific modeling task while at the same time keeping the model sufficiently simple, so that is practically useful.

[51] Oreskes et al. (1994) have even argued that verification of models of natural system is impossible. Furthermore, there is a large amount of literature discussing how verification and validation is done properly: i.e.: Oberkampf and Roy 2010; Roy 2005; Sargent 2013.

An additional complication for the representational function of the model arises if the target system consists of a large number of dynamically interacting individuals. Through self-organising processes, so called emergent phenomena can arise, which cannot be predicted from the behaviour of single organisms alone, but are a non-linear product of the agents’ interactions. This often puts limitations on the potential for generalisation and scaling of models because the emergent effects might appear or disappear if certain thresholds are passed.

What do models actually represent? We want to argue that in many cases there exists a discrepancy between the modelers goal and the results. The goal is that the model represents the target system and that processes that govern the phenomenon we are interested in are represented in the structures of the model. What happens however is that the modeler models the data that have been acquired about the target system and tries to fit the model as good as possible to this data even though the data are flawed and biased through the process of observation. We want to illustrate this discrepancy in the following chapter with an example from ecology before we discuss possible reactions.

4 Model building in ecology

To understand the specific relations between target system, data and models in theoretical ecology, we focus on the field of food web ecology.

Food webs describe network representations of feeding (“trophic”) interactions between species coexisting in defined habitats (Dunne 2009). The two main observables are species or groups of species and the feeding interactions between them, carrying energy, carbon, nutrient or other matter flow. Also in- and outflows to the environment and external compartments such as detritus can be included. Species sit at different positions in the food web, the so-called trophic levels. The bottom trophic level of a food web is comprised of autotrophs, which produce complex organic compounds from an external energy source such as light and simple inorganic molecules such as carbon dioxide. The higher trophic levels of food webs consist of heterotrophic organisms, which ingest organic carbon for energy production and biosynthesis. Food webs are generally represented in the language of graph theory and conceptualized as networks, where species are represented by vertices or nodes, and the trophic interactions by directed weighted edges or links. This allows embedding the field within the discipline of complex network analyses, using tools from network theory to quantify, analyse and model the structure and dynamics of food webs. Common described food web properties are for example the number of species and links in a food web, food web connectivity, the number of trophic levels, chains and cycles, clustering coefficients, degree distributions and degrees of nestedness and hierarchy (Dunne 2009).

4.1 Target system

The represented *target system* of food web ecology is the species community including all individuals and their feeding interactions. According to the UN Millennium Report, ecosystems are defined as "a dynamic complex of plants, animals and microorganism communities and the non-living environment, interacting as a functional unit. Humans are an integral part of ecosystems" (UNEP 2006). Complexity is hence an integral part of an ecosystem. Ecosystems are open systems with regards to energy, mass and information exchange, which means that any drawn system boundaries on the planet earth are arbitrary and need to be carefully assessed. Furthermore, ecosystems grow and develop, making them dynamic and any static description only a snapshot in time. Species abundances as well as their feeding preferences vary over time, depend on the current external and internal conditions of the ecological community and lack universality. Due to the interconnectivity of ecological networks, these dynamics include feedbacks, stochasticity and complex responses to disturbances, which lead to non-linear responses and impede predictions of detailed ecosystem behaviour (Jørgensen et al. 2007).

4.2 Data

The beginnings of food web ecology were merely descriptive studies of single food webs, aiming at increasing knowledge about species occurrences and their interactions at specific sites. The derivation of ecological theories about food web characteristics however needs to be based on larger data sets of different food webs, to be able to find common properties and fundamental mechanisms and patterns. Comparative food web studies have a short history (starting in the 1970s (Cohen 1977)), not least because of the difficulties connected with the translation of ecological communities into *data sets*.

The regarded key characteristics of ecological systems (openness, size, spanning of scales) make it close to impossible to gather exhaustive observation data of ecosystems. Furthermore, data collection in ecology is very labour- and time-intensive. Food webs need to be studied across seasons and over different years to account for environmental and weather variability. Complete species lists need to be compiled and feed interactions need to be derived based on observations, gut samples or expert interviews. More complete data sets can be deducted from less complex target systems such as model food webs in the lab. In the controlled environments, food webs can be kept very simple with only a few species and control over in- and outflows. This gain in completeness of data however comes at the cost of completeness of the target system: the model food webs are generally so low in degrees of complexity compared to real-world ecosystems, that conclusions and generalisations are not necessarily possible.

Grounding food web description in graph theory, ecologists have found a straightforward way to reduce much of the complexity of food webs. The basic observable is ‘species’. While in principle, biological species are defined genetically or by the capability to interbreed, some food webs explicitly lump species with the same predator and prey species into so called functional species. Depending on the questions that should be answered with the model, this can be appropriate and enhance the functionality of the resulting model. Very often however, the choice of species resolution follows no clearly defined protocol and is based on available observations. This regularly leads to the overrepresentation of better detectable species such as large mammals in contrast to small autotrophs or other animals with low body size. There is thus a bias in the resolution of data towards species, which are better detectable by the observer and the bias is strongly dependent on the scale of human perception.

Apart from the incompleteness and bias of food web data sets, another complication arises from the intrinsic dynamics of ecological communities. In reality, neither species abundances nor feeding preferences are static properties. Especially the latter are heavily dependent on the current configuration of the food web, the presence and absence of preferred prey species, the spatial distribution of individuals and further restrictions in prey availability and starving pressure. What is established as a feeding link in a food web representation is in fact only a propensity of a species to predate on another, which exceeds a certain threshold leading to an observation in the field. Most food webs which comparative food web analyses are based on are “cumulative” food webs, being time- or spatially averaged or cumulated (Dunne 2009). While this non-rigidness of presence or absence of interaction links between species questions the whole concept of a graph representation of a food web itself, it certainly has to be born in mind, that current theoretical food web ecology analyses is only a subset of the potentially possible, but an average of the at any moment actually occurring species interactions.

4.3 Model

Ecological models are designed and explored to reproduce patterns, which are found in observations of natural ecosystems. Observational patterns in food web data are for example the relationship between the number of species and the number of interaction links, the distribution of feeding links across species and species across trophic levels, the clustering of species and their centrality in the food web as well as characteristics of species interactions, matter transfer and cycling. These numbers, ratios or functions are analysed across datasets of different food webs for general unifying patterns. In parallel, theoretical ecologist hypothesize the main mechanisms and processes driving ecosystems, develop according models and attempt to recreate the observed patterns with their models by solving differential equations or running computer simulations. Should the patterns be found both to be consistent across spatial and temporal scales and to be regenerable by mechanistic models, the described patterns are elevated to the status of a theory.

It has for example been assumed for a long time, that connectivity[52] must be linked to key survival criteria of ecosystems such as stability, resilience and functionality. This is the reason why the relationship between links and species has been of long-standing interest. Early comparative food web research and theoretical stability considerations hypothesized a constant ratio of the number of links L and species S ($L = a*S$). In studies of about 130 food webs, a constant relation of $L/S \sim 2$ has been found and backed up by different food web assembly models (Dunne 2006) – a theory was about to come into being. However more

[52] In this context, connectivity refers to the ratio between the number of species and the number of feeding relationships between species in the ecosystem.

comprehensive data sets shifted this notion of generality as they rather suggested relationships of $L \sim S^2$. More data, more studies and reanalyses of old data sets complicated the situation further, reporting exponents a ($L \sim S^a$) ranging between ⅔ and 2. More recent models therefore focus on finding mechanisms which can reproduce different exponents based on external criteria, potentially finally establishing $L \sim S^{x(\text{conditions})}$ as a theory (Dunne 2006).

From these considerations behind one of the most basic applications of models and theories in ecology, it becomes obvious, how strongly ecological theories depend on the choice of species and species interactions. Depending on the included links and the aggregation of species, completely different statistical patterns emerge. The sensitivity of different food web theories to data quality and resolution varies between target food web properties (e.g. connectance, food chain lengths, number of species per trophic level), but all food web theories are affected (Dunne 2006).

As discussed in the previous section, any models in data-driven sciences aim to explain phenomena in the actual target system, but in fact describe and assess observations or data of the target system, and ecological models are no exception. Over the last decades with the improvement of ecological field methods and increased scientific interest in ecosystem functionality, food web data resolution has increased tremendously. Nevertheless, the initially discussed shortcomings of ecological data (incompleteness, bias and disregard of dynamics) could not be resolved yet. In the following, we will discuss the implications and offer three possible solutions for the discrepancy.

5 Conclusion & Outlook

Models are a central epistemic tool in theoretical ecology and are used with the goal to represent, explain and predict behaviour of phenomena in target systems. In this paper, we highlighted a number of problems the use of models in theoretical ecology has to face. Data about the phenomena are required for building and validating the models. The data in theoretical ecology is biased, incomplete and disregards certain dynamics in the target system leading to known and unknown errors and misrepresentations of the phenomena. We argue that the bias and many of the known errors in the data occur due the choice of species as observables that led to a very fine-grained and detailed description of organisms that are easily accessible due to their size and habitat in comparison to a more coarse classification of less accessible organisms. Species aggregation in practice very often depends on the accessibility of the target systems. The flawed data is used to find patterns and regularities that in the absence of a detailed theory constitute the basis for the models. This leads to the situation that the model is a model of the data rather than a model of the phenomenon and its explanations and predictions are therefore contrary to the original goal also targeted on data of the phenomenon.

We can imagine three different way to address these problems in theoretical ecology.

The first path would be to accept the limitations, which come with the choice of species as observable and continue to work with data of phenomena the way it is done now rather than try other ways to learn more about the phenomena themselves. The consequence would be to operate based on flawed data consciously. With the collection of more data, the models can be improved further and better explanations and predictions for the data can be given. This has been a successful method in theoretical ecology and could continue this way. The representational function of models in theoretical ecology is limited to the data of the phenomena in this case and expectations and goals have to be formulated accordingly.

The second path is a more radical one: to discard species as the main observable since too many biases and flaws in the data follow from this choice. What could alternative observables be? The goal has to be that observables can be observed and described equally well on all scales. Finding such a new observable and establishing it is a very challenging task. If successful, this would however eliminate one major reason for flaws in ecological data. One first try in this direction has been made at the intersection of theoretical physics and microbial biology. Here mathematical frameworks are under development, which attempt to describe a more continuous genotype space, eradicating the dependence on clearly defined species for theory development (Tikhonov 2017).

The third path follows the goal of the second one but does not give up species as the main observable. Developments in gene-sequencing and big data analysis enable ecologists to gather very precise data for species of all sizes. New types and amounts of data will be available for the classification of species. Observational data, which are generated through these methods can be equally precise on all scales and could avoid the bias of current data, which has its source in the current preference for human-scale organisms. The challenge for the theoretical ecologists is to come up with a definition of species, which makes use of this observational data to distinguish between species.

Models in theoretical ecology offer plenty of opportunities for philosophers of science to find fruitful examples and research questions. More cooperation between the two fields would not only be beneficial for ecologist but might also offer new insights in epistemological questions about models and their representational function.

References

Arakawa, Akio (1966). Computational Design for Long-Term Numerical Integration of the Equations of Fluid Motion: Two-Dimensional Incompressible Flow. Part I. *Journal of Computational Physics* 1: 119–143.

Bailer-Jones, D. M. (2009). *Scientific models in philosophy of science*. University of Pittsburgh Pre.

Begon, M., & Townsend, C. R. (2021). *Ecology: from individuals to ecosystems*. John Wiley & Sons.

Bogen, J. (2017). Theory and Observation in Science. *The Stanford Encyclopedia of Philosophy*. Zalta, E. (ed.).

Bogen, J., & Woodward, J. (1988). Saving the phenomena. *The philosophical review*, *97*(3), 303-352.

Cartwright, N. (1983). *How the laws of physics lie.* Oxford.

Cohen, J. E. (1977). Ratio of prey to predators in community food webs. *Nature*, 270(5633), 165-167.

Courchamp, F., Dunne, J. A., Le Maho, Y., May, R. M., Thébaud, C., & Hochberg, M. E. (2015). Fundamental ecology is fundamental. *Trends in ecology & evolution*, *30*(1), 9-16.

Dunne, J. A. (2006). The network structure of food webs. In M. Pascual & J. A. Dunne (Eds.), *Ecological networks: linking structure to dynamics in food webs* (pp. 27–86). New York: Oxford University Press.

Dunne, J. A. (2009). Food Webs. *Encyclopedia of Complexity and Systems Science*, 3661–3682.

Frigg, R., & Hartmann, S. (2020). Models in Science. *The Stanford Encyclopedia of Philosophy.* Zalta, E. (ed.).

Frigg, R. & Nguyen, J. (2017). Models and Representation. in: Magnani, L. & Bertolotti, T. (eds.). *Springer Handbook of Model-Based Science*. Springer, Cham.: 49-102.

Giere, R. (2004). How Models Are Used to Represent Reality. *Philosophy of Science 71(5)*: 742–752.

Hastings, A., & Gross, L. (eds.). (2012). *Encyclopedia of theoretical ecology* (No. 4). University of California Press.

Hui, D. (2012). Food web: concept and applications. *Nature Education Knowledge,* 3(12), 6.

Jørgensen, S. E., Fath, B. D., Bastianoni, S., Marques, J. C., Muller, F., Nielsen, S. N., ... & Ulanowicz, R. E. (2011). *A new ecology: systems perspective*. Elsevier.

Lenhard, J. (2007). Computer simulation: The cooperation between experimenting and modeling. *Philosophy of Science*, *74*(2), 176-194.

Lorenz, E. N. (1963). Deterministic nonperiodic flow. *Journal of atmospheric sciences*, *20*(2), 130-141.

Marquet, P. A., Allen, A. P., Brown, J. H., Dunne, J. A., Enquist, B. J., Gillooly, J. F., … West, G. B. (2014). On theory in ecology. *BioScience*, *64*(8), 701–710. https://doi.org/10.1093/biosci/biu098

Mazzocchi, F. (2015). Could Big Data be the end of theory in science? A few remarks on the epistemology of data-driven science. *EMBO reports*, *16*(10), 1250-1255.

Morgan, M. (1999). Learning from Models. in: Morgan, M. & Morrison, M. (eds.). *Models as Mediators: Perspectives on Natural and Social Science*. Cambridge: Cambridge University Press: 347-388. doi:10.1017/CBO9780511660108.013

Morgan, M. & Morrison, M. (eds.) (1999). *Models as Mediators: Perspectives on Natural and Social Science*. Cambridge: Cambridge University Press.

Oberkampf, W. L., & Roy, C. J. (2010). *Verification and validation in scientific computing*. Cambridge University Press.

Oreskes, N., Shrader-Frechette, K., & Belitz, K. (1994). Verification, validation, and confirmation of numerical models in the earth sciences. *Science*, *263*(5147), 641-646.

Proulx, R. (2007). Ecological complexity for unifying ecological theory across scales: A field ecologist's perspective. *Ecological complexity*, *4*(3), 85-92.

Roy, C. J. (2005). Review of code and solution verification procedures for computational simulation. *Journal of Computational Physics*, *205*(1), 131-156.

Sargent, R. G. (2013). Verification and validation of simulation models. *Journal of simulation*, *7*(1), 12-24.

Scheiner, S. M., & Willig, M. R. (2008). A general theory of ecology. *Theoretical Ecology*, *1*(1), 21–28. https://doi.org/10.1007/s12080-007-0002-0

Tikhonov, M. (2017). Theoretical microbial ecology without species. *Physical Review E*, *96*(3), 32410. https://doi.org/10.1103/PhysRevE.96.032410

Turner, M., Dale, V., & Gardner, R. (1989). Predicting across scales: Theory development and testing. *Landscape Ecology*, *3*(3), 245–252. https://doi.org/10.1007/BF00131542

UNEP (2006). Marine and Coastal Ecosystems and Human Well-Being: A Synthesis Report Based on the Findings of the Millennium Ecosystem Assessment. *Nairobi, Kenya: Environmental Programme.*

Williams, R. J. (2010). Simple MaxEnt models explain food web degree distributions. *Theoretical Ecology*, *3*(1), 45-52.

Chapter 5: Computer simulations, machine learning and the Laplacean demon: Opacity in the case of high energy physics

1 Introduction

High energy physics (HEP), the study of the scattering of material particles at high energies, is a field of research in which the use of computer simulations (CSs) abounds, a fact that has attracted the attention of a bunch of philosophers (Karaca, 2017; Massimi and Bhimji, 2015; Morrison, 2015). In recent years, machine learning (ML) techniques have become more and more important in HEP as well, e.g. for the discrimination of signal from background data. Even more recently, the HEP community has begun to explicitly acknowledge the "black box"-nature of ML (Chang et al., 2018, p. 1), i.e., the *opacity* associated with it. CS and ML (including supervised techniques, on which we focus on in this paper) share methodological features as well as epistemological challenges. Both can be used for the exact same purposes (recognition and prediction of statistical patterns) and both raise concerns of opacity, uncertainty, and robustness. This makes HEP a natural study case for investigating philosophical concerns of opacity associated with both CSs and ML in modern science.

In this paper, we will pursue the following three general aims: (I) We will define a notion of *fundamental* opacity and ask whether it can be found in HEP, given the involvement of ML and CS. (II) We will identify two kinds of non-fundamental, *contingent* opacity associated with CS and ML in HEP respectively, and ask whether, and if so how, they may be overcome or at least greatly reduced. (III) We will address the question of whether any kind of opacity, contingent or fundamental, is unique to ML or CS, or whether they stand in continuity to kinds of opacity associated with other scientific research. In short, we will argue that (I) there *is* a fundamental kind of opacity, that (II) contingent kinds *can* be overcome in HEP, to a certain extent, and that (III) all identified kinds of opacity are *continuous* with opacity in non-computer aided science. In (II) we will also establish that even though the two opacities associated with CS and ML are both contingent, the origins and nature of the respective opacities are significantly different. As a consequence, there are differences in *how* to overcome the contingent opacity associated with CS and ML respectively. Notably, in the overcoming of contingent opacities associated with CSs we will focus closely on HEP, as it presents a field of research in which this is done exceptionally well, whereas the opacities to be overcome in ML are a current hot topic for the ML community, not specifically for physics.

1.1 Computer simulations in high energy physics in a nutshell

CSs[53] are used in HEP for various purposes, ranging from the design of equipment or the guidance of searches and analyses to the estimation of expected 'backgrounds' in experiments, i.e., measurable activity in detector and '(particle) beam pipe' to be expected in the absence of phenomena of interest. For concreteness, let us briefly zoom in a little more detail on the complex simulation infrastructure used by CERN's ATLAS experiment.[54] Here, as in the other experiments at the Large Hadron Collider (LHC), different stages of a collision event and subsequent happenings[55] will be overall separated into *event generation* and *detector simulation*. Event generators then again factor into *hard process*, the highest-momentum scattering between two constituent 'partons' (quarks or gluons) inside the protons, *parton shower*, the production of a large number of additional partons produced in subsequent decay and radiation events, *hadronization*, the formation of complex particles ('hadrons') out of this shower of partons, and the *decay* of hadrons not stable enough to reach the detector. In addition, an *underlying event* may be simulated, meaning a second comparatively high-energetic interaction between two further partons which leads to added activity in parton shower, hadronization, and decay.

The detector simulation then uses all stable particles from the final stages as input and simulate their interaction with the real-world detector, as well as known background fluctuations. This is done based on a vast range of models coming from atomic, solid state, nuclear, and particle physics respectively, which will sometimes be largely based on theory, sometimes rather phenomenological, in the sense of being somewhat independent from theory in construction (Suárez and Cartwright, 2008, p. 70), and sometimes just collections of parametrized data.[56] The main detector simulation software used at ATLAS (GEANT 4) also includes a model of the geometry, in order to adequately simulate the spatial distribution of these interactions. There is also the possibility of choosing specific physical conditions of the detector, to test the impact of these on one's experiment.

[53] We will here not offer a general account of what counts as a CS, since this has been done in various places in the philosophical literature. The reader may be referred e.g. to Humphreys (2004, Chap. 4) instead. Boge (2019) also offers some insights.

[54] E.g. ATLAS (2010). For a detailed presentation in a philosophical context cf. also Boge and Zeitnitz (2020).

[55] In fact, whole 'bunches' of 100,000 million protons, with a bunch being of the size of 64 microns, will travel in two beams accelerated to 7 TeV inside the LHC. These will cross every 25 ns which results in around 600 million collisions per second (cf. https://lhc-machine-outreach.web.cern.ch/lhc-machine-outreach/collisions.htm; checked 11/18).

[56] To get an impression, one may consult the GEANT physics reference manual (cf. GEANT Collaboration, 2016).

1.2 Deep learning in a nutshell

ML techniques are today involved in many areas of science and social life, with applications ranging from risk management of water distribution systems or the prediction of severe weather conditions to more straightforward recognition and classification tasks (e.g. Baldi et al., 2014; Goodfellow et al., 2016; Rudin and Wagstaff, 2014). Deep learning (DL) is a specific kind of (ML) using *connectionist* (or 'neural') *networks*, in which the network includes multiple hidden layers (the number of which defines its *depth*).

A neural network can be understood as an arrangement of nodes (or 'neurons') which are ordered into different layers. The neurons of one layer all execute the same (typically non-linear) function on their input and then hand the result over to one or multiple nodes of the next layer.[57] A standard way to represent these relations is by appeal to directed graphs, wherein the nodes correspond to labeled vertices and the input-output relation corresponds to directed edges (cf. also Figure 6). If there is no previous or next layer, the layer in question will be an input or output layer respectively. All other layers are called 'hidden'.

Let's consider a simple example.[58] Assume that one's network were supposed to learn the exclusive-or function XOR: $\{0,1\}^2 \rightarrow \{0,1\}$, which gives 1 for the pairs (0,1), (1,0) and 0 else. Such a network could be mathematically represented as taking an input vector $x \in \{0,1\}^2$ and as successively transforming it by applying a sequence of functions. Each entry $x_i$of x would then correspond to an input node, and the successively executed functions would correspond to hidden- or output layers respectively.

A simple network capable of performing this task can be constructed with only one hidden layer that executes a vector valued function

$$\boldsymbol{h}(\boldsymbol{x}; \boldsymbol{W}, \boldsymbol{c}) = \boldsymbol{W}\boldsymbol{x} + \boldsymbol{c}. \qquad (1)$$

The letter 'h' here indicates that the layer is hidden, W represents a matrix of weights that the inputs receive, and c a vector of biases that may output some (non-zero) value even if s = 0. Because of the nature of matrix multiplication, both inputs may contribute to both nodes (h_i) of

[57] Strictly speaking, this is only an ideal limit; a network may, for instance, have "skip connections" which jump from layer *i* to layer *i*+2 (Goodfellow et al., 2016, p. 196). We here also eschew the discussion of other kinds of networks, such as recurrent ones, which may not be organized into proper layers at all. Our discussion will be restricted to *feedforward* networks.

[58] Cf. Goodfellow et al. (2016, p. 166) for the following, which is our main reference on ML/DL here.

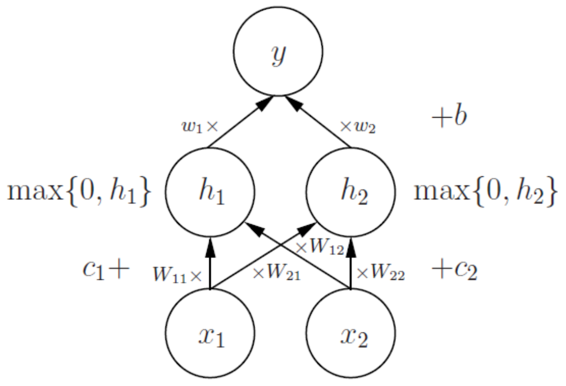

Figure 6: Representation of the network described below by a directed graph. The edges may be interpreted as feeding forward a weighted and biased version of outputs received from a set of 'parent' nodes (vertices). Each 'hidden' vertex may be thought of as computing an activation function on its input. (Reproduced with added detail from Goodfellow et al., 2016, p. 169.)

the hidden layer, depending on the weights w_{ij} contained in W. If the output layer now executes the function

$$f(x; W, c, w, b) = w \cdot \max\{0, h\} + b, \qquad (2)$$

where w and b are a vector and scalar with analogous roles as W and c, $\cdot$ represents the standard scalar product on $\mathbb{R}^2$, and the max-function is applied component-wise, there is a choice of all the *parameters* (the quantities specified behind the semicola of h and f) such that the network performs the specified task. Establishing that choice through an iterative, adaptive procedure (see below) may be called *learning*.

A function executed by a hidden layer is generally called an *activation function*, and these are often times non-linear. In many cases these are sigmoidal functions, which "ensures there is always a strong gradient whenever the model has the wrong answer." (Goodfellow et al., 2016, p. 177) $g(z) = \max\{0, z\}$ is another "default choice" (ibid., p. 186) called the *rectified linear unit* that allows for several generalizations (cf. ibid., p. 187). All in all the network may be said to execute the composite function $f \circ h$ on $\{0,1\}^2$, which correctly represents XOR for a certain choice of the parameters.

How is this choice of parameters, sometimes collectively denoted by θ, achieved? The standard method is to let the network optimize a *loss* or *cost function* over values of θ, i.e. a function that specifies how far the network's output is from a desired result. The error defined by this loss function is the so called *training error*, which occurs during the stage where the network is confronted with preselected data on which it is tuned for fulfilling the specified task. Neural networks also face the danger of a *generalization error*, which refers to the "expected value of the error on a *new* input." (Goodfellow et al., 2016, p. 107; emph. added) Striking a balance between over- and underfitting, i.e. being too specific or not specific enough to the data the network is trained on, is one major practical task one faces in the design-stage of DL algorithms (ibid., pp. 108 ff.).

Quite usually, optimization is performed by using iterative, *gradient-based* methods, where one 'walks along' the negative gradient in predefined steps.[59] The most important instance certainly is *stochastic gradient descent*, where one updates θ in small steps $\theta \to \theta' = \theta - \epsilon g$, with

$$g = \frac{1}{m} \nabla_\theta \sum_{i=1}^{m} - \ln p\left(y^{(i)} \middle| x^{(i)}; \theta\right). \quad (3)$$

Here $p\left(y^{(i)} \middle| x^{(i)}; \theta\right)$defines the likelihood of output $y^{(i)}$ given input $x^{(i)}$(parametrized by θ), ϵ is a predefined step size also called the *learning rate*,[60] and *m* is either the cardinality of the *training set*, the set of data which the network encounters (and is configured on) before it is put to its actual task, or that of a 'minbatch', a (comparatively small) subset of examples form the (possibly huge) training set. In order to strike a balance between over- and underfitting, it may be useful to add a regularizer $\lambda\Omega(\theta)$ to the loss function, with λ a tunable parameter. Stochastic gradient descent effectively means approximating the value of θ for which the obtaining of $x^{(i)}$ makes the corresponding output $y^{(i)}$ most likely.

The gradient may here be computed with the *backpropagation algorithm*, which "allows the information from the cost to [...] flow backward through the network" (Goodfellow et al., 2016, p. 197), by computing for each node its partial derivatives w.r.t. parent variables and combining them by an appropriate chain rule. This includes observing possible changes of the network's output under changes of the weights. Insofar as the optimization task involved in (stochastic) gradient descent is executed exactly by adjusting these weights (cf. Rumelhart et al., 1986), the algorithm lends a certain autonomy to the network: it can adapt its own infrastructure to perform better at a particular task, i.e., 'learn' a successful performance.

The network discussed above is not really deep but rather 'shallow', as it only has one hidden layer. Deep networks (DNs) used to suffer from a so-called *vanishing gradient problem* in the past, which can be understood roughly as follows: Assume that the network repeatedly multiplies a vector of data by the weight matrix *W* .[61] In case *W* has *n* linearly independent eigenvectors $v^{(j)}$with eigenvalues λ_j respectively, we can write $W = V\mathrm{diag}(\lambda_j)V^{-1}$, where *V* is a matrix with the $v^{(j)}$ as columns and inverse V^{-1}, and $V\mathrm{diag}(\lambda_j)$ a diagonal matrix with the eigenvalues on

[59] Recall that the gradient ∇f of some function f points in the direction of f 's steepest ascent, so moving along $-\nabla f$ will get one in the right direction. Of course one may instead hit a saddle point in this way, so finding (local) minima strictly requires investigating the Hessian as well.
[60] Some algorithms make the steps size a function of prior steps, e.g. by scaling each component by the inverse absolute value at the prior step, so that descent in steep/shallow directions will decrease faster/slower.
[61] This is of course a quite restrictive assumption, but if the weights are similar enough, the problem will arise in similar ways.

its main diagonal. After m repeated applications of W, the effect will be $(V\text{diag}(\lambda_j)V^{-1})^m = V\text{diag}(\lambda_j^m)V^{-1}$, which will vanish quickly in case the λ_j are smaller than 1. When computed by the chain rule, however, the gradient can equally be seen to scale with the weights of previous layers; so the learning may get stuck in early layers of a DN. Today, there is a number of ways to handle this problem, e.g. training the network on simpler tasks first, or even training a simpler network that is made more complex once the simpler one has successfully learned. Cf. Baldi et al. (2014, p. 2) and references therein for further examples.

Moreover, it is not always convenient to specify a *fixed* set or even a fixed number of parameters, whence some models incorporate *non-parametric methods* where no such restriction is imposed. An example is *nearest neighbor regression*, where the network classifies data points *x* by associating them with entries from the training set that have the minimal distance to *x* in some chosen norm.

Finally, in the task described above the examples encountered in the training stage are associated with *targets*, i.e., specific outputs *y* the network is supposed to achieve, when confronted with a respective *x*. Similarly, a network may be required to learn to *predict* a specific target *y* from an input *x*, usually by learning a conditional distribution $p(y|x)$. Algorithms of this kind are sometimes referred to as *supervised*, whereas algorithms that merely gather information about the distribution $p(x)$ of a set of data *s* are called *unsupervised*. Supervised applications include, for instance, classification tasks, regression tasks (essentially: the prediction of a value), or transcription tasks in which a relatively unstructured representation of relevant data is transcribed into textual form. Unsupervised learning tasks include probability density estimation, learning to draw samples from a distribution, or denoising data.

Although it is not generally known which nodes represent what, neural nets imply the possibility of a *distributed representation*, meaning that different sets of nodes can become specialized to processing different features of an object. For instance, in describing different types of vehicles with different colors one set may represent the values of the variable 'color' whereas another may represent the values of the variable 'vehicle type'. Similarly, when representing a group of people, one set may handle person identities whereas another their relationships (cf. Rumelhart et al., 1986, p. 534). The output layer will then indicate the learned relationships, and the network may be said to represent them as distributed patterns of activity in the nodes. Factoring inputs in such a way can have several advantages, particularly when it comes to computational cost and flexibility.

1.3 The rise of deep learning in high energy physics

As can be seen from this brief discussion already, the ML community today has developed a wide range of methods, tuned to specific tasks. Indeed, ML techniques have long been used in *high energy physics* (HEP), but have here received increased interest in recent years, due to the advances in handling vanishing gradients and the associated performance increase of DNs. Uses of ML in HEP include the identification of jet-substructures, which can be used to probe the Standard Model (SM) and increase sensitivity in finding evidence for physics beyond the SM (Larkoski et al., 2017), uncertainty estimation and combination, or the determination of parton density and fragmentation functions (Carrazza, 2018). Maybe the most important HEP application is, however, in the (ongoing) search for *new* particles (cf. Baldi et al., 2014, p. 2). In such a search, physicists need to “isolate a subspace of their high-dimensional data in which the hypothesis of a new particle or force gives a significantly different prediction than the null hypothesis, allowing for an effective statistical test.” (ibid., p. 1)

The crucial quantity for significance evaluation here is the *likelihood ratio*, i.e., the ratio $p(D|\Theta)/p(D|\Theta_0)$, where D represents the data, Θ the hypothesis to be tested, and Θ_0 a null hypothesis. In the case of searches for exotic particles in LHC’s proton/proton scattering, these take the generic form of scatterings and decay-chains:

Θ: *pp* → ‘known debris’ + (intermediate) exotic particle(s)

Θ_0: *pp* → ‘known debris’

For the purposes of experiment, both Θ and Θ_0 have to be translated into suitably connected hypotheses about a *test statistic*, such as total mass or (missing) transverse momentum (cf. James, 2006, p. 255, and below). The two steps indicated in Θ andΘ_0, moreover, may themselves be intermediate steps in a larger chain of decays that involves only known (and measurable) ‘debris’ as final state products. These are typically *jets*, meaning conical distributions of hadronic particles, *leptons* such as electrons or muons, together with missing transverse momentum from corresponding (anti-)*neutrinos*, and *photons*, all of which will be registered by appropriate parts of the detector in use.

Quite often, however, the relevant likelihoods “cannot be expressed analytically, so simulated collision data generated with Monte Carlo methods are used as a basis for approximation of the likelihood function.” (Baldi et al., 2014, p. 1) Still, the “high dimensionality of data [...] makes it intractable to generate enough simulated collisions to describe the relative likelihood in the full feature space”. (ibid.)

Statistical hypothesis testing may be seen as kind of classification task: if a certain set of parameters exceeds certain bounds, we call our observations 'significant' and think of them as possibly indicating the presence of some phenomenon of interest. In the same way, an ML classifier may take "a complicated, high-dimensional input *s* and summarize it with a single category identity *y*." (Goodfellow et al., 2016, p. 233) Hence physicists in HEP use ML classifiers as a means of 'dimensionality reduction': a set of criteria is used to define a hypersurface in the feature space beyond which data will count as signal, and below (or on) which they will count as 'background'. This hypersurface then defines a cut-value for a suitable test statistic in hypothesis testing, which equally defines the cut-off between the two classes of the classifier.

In learning to classify correctly, the network may be said to effectively learn to *represent* the corresponding likelihood functions. This should be compared to the toy-example discussed in section 1.2, where the network may equally be said to represent the XOR function in terms of the functions executed by the individual layers, when the parameters take on appropriate values. One should caution against reading too much into this notion of 'representation', however, since it does not involve any implication of *intentionality* that one might intuitively associate with the term. We will return to this issue below.

So ML classifiers in HEP searches can learn to distinguish 'signal' from 'background' data. But what exactly does that mean? By 'background', one means any kind of data present in the relevant region in which distributions of quantities of interest are measured, but which is not connected to the process of interest. For instance, this could be data with identical (measurable) final state products, but where the distributions of kinematical features will be different from those in the signal data, because of the occurrence of exotic particles as intermediate states in the latter case and their absence in the former. Baldi et al. (2014), in fact, considered two benchmark cases of exactly this kind in order to evaluate the performance of a DN with five hidden layers as a classifier against that of a shallow one with only one hidden layer.

One of these benchmark cases, which we shall discuss in a little more detail, concerned processes that would involve new theoretical Higgs bosons. As is well known, a 'light' Higgs boson, compatible with the predictions of the SM of particle physics, has been found in 2012 by ATLAS and CMS (cf. Aad et al., 2012; Chatrchyan et al., 2012). However, it is a theoretical possibility that there exists a more massive electrically neutral Higgs boson, H^0, which decays into the known 'light' Higgs boson, h^0, via (positively or negatively) electrically-charged Higgs bosons, $H^\pm$. More precisely, a set of possible decay chains would be

$$gg \to H^0 \to W^{\mp}H^{\pm} \to W^{\mp}W^{\pm}h^0 \to W^{\mp}W^{\pm}b\bar{b}, \qquad (\Theta)$$

where g are gluons contained in the scattering protons, $W^{\pm}$ are electrically charged intermediate vector bosons, and b, $\bar{b}$ are (anti-)quarks known to produce characteristically broad jets (e.g. Llorente and Cantero, 2014). This was to be contrasted with background processes of the form

$$gg \to g \to t\bar{t} \to W^{\mp}W^{\pm}b\bar{b}\,, \qquad (\Theta_0)$$

with *t* the weak isospin-partner[62] of *b*, in which the heavier Higgs's would not be present, but where the final-state products would be the same. Since neither *W*s nor *b*s are directly detectable, one would have to make a selection on the decay modes of these. The selection in place was on events with one lepton (ℓ) and missing energy from the corresponding neutrino[63] ($\mathcal{v}$) from the decay of one of the *W*s, two jets from the decay of the other one (jj), and two further ones from the decay of the *b*s (*bb*). These were required to satisfy the following characteristics: (i) Exactly one electron or muon, with transverse momentum $p_T > 20$ GeV and pseudorapidity satisfying $|\eta| < 2.5$;[64] (ii) at least four jets, each with $p_T > 20$ GeV and $|\eta| < 2.5$; and (iii) jets that could be 'tagged' as resulting from *b*s rather than gluons or lighter quarks. Transverse momenta, the reconstructed missing transverse momentum for the neutrino, as well as the '*b*-tagging' information were regarded as 'low-level' features, since transverse momenta are more or less directly inferred from electric activities in the detector.

Process (Θ) could, however, also be associated with rather different hypotheses regarding *higher-level* features than process (Θ_0), meaning features that are not measured or directly inferred from measured quantities (like missing momentum) but rather defined as (typically nonlinear) functions thereof. An example are reconstructed invariant masses, where the invariant mass of particle 1, decaying into (detectable) particles 2 and 3, for instance, is reconstructed in natural units as $m_1^2 = m_{23}^2 = (E_2 + E_3)^2 - |p_2 + p_3|^2$, due to the relativistic energy-momentum relation and conservation laws. According to both, (Θ) and (Θ_0), there should be peaks in the $m_{\ell\mathcal{v}}$ and m_{jj} distributions at the known mass m_W . However, (Θ_0) would predict

[62] The weak isospin is a concept similar to the isospin, which was introduced to understand the many similarities between protons and neutrons (up to their charge).

[63] Recall that neutrinos are only detectable through very subtle measurements. Hence, one usually infers their existence rather from ('missing') energy and momentum not measured but expected on account of conservation laws.

[64] $p_T = \sqrt{p_x^2 + p_y^2}$ is the transverse momentum, i.e., the momentum component particles carry in the plane transversal to the beam direction, where $(p_x, p_y, p_z)^t$is the particle's three-momentum in a laboratory frame and p_z lies in the beam direction. The pseudorapidity η is defined as $-\ln\tan(\frac{\theta}{2})$, where θ is the polar angle of a right handed coordinate system with origin at the interaction point.

peaks in $m_{j\ell\nu}$ and m_{jbb} distributions at the known value for m_t, whereas (Θ) would predict peaks at known m^0 in m_{bb} as well as at $m_{H^\pm}$ in m_{Wbb} and at m_{H^0} in m_{WWbb} respectively, for assumed masses $m_{H^\pm}$ = 325 GeV and m_{H^0} = 425 GeV of $H^\pm$ and H^0 respectively.

Lower level features usually have a limited *discrimination power*: Often the p_T-distributions corresponding to signal and background hypotheses differ only slightly, so classification according to these features is difficult. Shallower networks are hence usually trained on higher-level features, and performance can be further increased if both high- and low-level features are available to them.

Baldi et al. (2014) now compared the performance of shallow and deep networks, where low-level features were used as inputs for the DNs and higher-level ones for the shallow networks. To assess the advantage of adding hidden layers, both kinds of networks were trained with the same number of hidden units and the same *hyper-parameters*, i.e., parameters of the network to be determined in an optimization procedure outside the training for the actual task. Different classifiers were trained on low-, high-, or combined features respectively, in order to assess which training-strategy would yield the best performance. They each encountered 2.6 million training examples from Monte Carlo data, and the training was validated on 100,000 further ones. The performance was then tested on 500,000 simulated events (ibid, p. 5).

In addition, performance was compared to *boosted decision trees*, where decision trees are "tree- structured classifiers that consist of a series of binary splits", effectively dividing "the multi-dimensional observable space into many (rectangular) volumes that are attributed to either signal or background." (Voss, 2013, p. 178) Boosting these means consecutively defining additional trees by increasing the weights on misclassified events and obtaining the final classification as a weighted average of outputs from all these trees. This is done to compensate individual trees' sensitivity to statistical fluctuations in training data, and has lead to "dramatic performance increases." (ibid., p. 179)

DNs fed with low-level inputs now significantly outperformed both boosted decision trees and shallow networks fed with the full feature set on two significance metrics: the discovery significance and the area under the receiver operating characteristic curve (AUC), where the former is the well known measure for the probability of false positives (high discovery significance = low probability), and the latter a joint measure for the probability of correct positives and negatives under changes of a cut-value on the test statistic (large area under the curve meaning a large range of cut-values for which both quantities are high; e.g. Voss, 2013, for details). DNs trained with the full feature set performed even slightly better, but those trained only on the high-level features surprisingly performed worse. The increase in performance upon adding the high-level feature was always less than the difference between the performance of DNs trained with low-level data and that of shallow ones or BDTs trained on high-level data.

The upshot is that while boosted decision trees and shallow networks are already very useful tools when fed with adequately predefined variables, handing over the task of discovering physical connections between measured (or immediately inferred) quantities to a DN will lead to an even better performance. In other words: increased success in using ML techniques comes at the price of giving up more (immediate) insight into the relations between physical variables.

Despite this remarkable usefulness of ML techniques for HEP, physicists have recognized that they essentially

> function as a *black box*: send in some data and out comes a number. While this kind of nonparametric estimation can be extremely useful, a physicist often wants to understand what aspect of the input data yields the discriminating power, in order to learn/confirm the underlying physics or to account for their systematics. (Chang et al., 2018, p. 1; emph. added)

A different way of putting things is that ML is *epistemically opaque*: some amount of insight into what is going on is not being made available to any user. To delineate the *nature* of this opacity in more detail however, and to what extent or in which sense it may or may not be overcome, we will now first turn to the philosophical debate, which has mostly focused on *computer simulations* (CSs). We will then define the unique sense in which ML or DL is opaque, beyond the sense already present in CSs, and subsequently discuss the (in)eliminability of both kinds of opacity.

2 Opacity in computer simulation and deep learning

What could it mean that ML, and particularly DL, is effectively a ‘black box’, i.e., epistemically opaque? Intuitively, we might refer to any kind of procedure as epistemically opaque if we cannot fully comprehend how a certain input leads to a certain output. Indeed, several authors in the philosophical literature (e.g. Humphreys, 2004, 2011; Kaminski, 2018; Lenhard, 2011; Saam, 2017) have claimed that this precisely applies to CSs.

From a certain vantage point, however, this seems counterintuitive: the structures involved in CSs are all *algorithmic*, and hence fully determined by the program code. For every step *n* it is predefined how to get to step *n* + 1, and all the rules are written down and can be reviewed by any agent who has access to the code. Following Lenhard (2011, p. 137), we may refer to this as the *algorithmic transparency* of CSs. The very same may be said, e.g., about a neural network: the execution of functions and optimization of parameters will be implemented as an algorithm, so it is transparent in the same sense. How could CSs and ML create results in an epistemically opaque way in spite of their algorithmic transparency?

A crucial first step to seeing this is to realize that opacity is intuitively relative to a (set of) cognitive agent(s): *we* do not understand the input-output relation in a certain procedure. This is reflected also in the classic definition by Humphreys (2011, p. 139):

> a process is epistemically opaque relative to a cognitive agent *X* at time *t* just in case *X* does not know at *t* all of the epistemically relevant elements of the process. A process is essentially epistemically opaque to *X* if and only if it is impossible, given the nature of *X*, for *X* to know all of the epistemically relevant elements of the process.

Using this definition, it is clear that many complex CSs come out as *prima facie* epistemically opaque. In CSs used by CERN's ATLAS experiment at the LHC, for instance, no human being will presumably ever be able to review all the steps of the computer program by herself, as O(10^6) lines of code, written in different programming languages, can be relevant for the full simulation of events and detection (cf. ATLAS, 2010, p. 835).[65]

Saam (2017, p. 79), moreover, points out that "while the program code (with several tens of thousands of lines of code) may be epistemically transparent, the compiled code is not (several million lines of code are possible)", where by 'compiled code' one means the translation of the program code into machine code, i.e., into a set of instructions the computer can actually execute (e.g. Clements, 2006, p. 205). Hence compilation may increase the amount of code by two orders of magnitude, so that the code instructing the machines used for CS at ATLAS may require up to O(10^8) lines.

[65] This code is managed by the Athena framework, which allows users to specify properties of the intended simulation through the JobOptions service, but also to some extent in the form of additional Python scripting (cf. ATLAS Computing Group, 2005, Sect. 3.3).

However, what about the notorious Laplacean demon, who would not be subject to human restrictions? For him, studying even complied code would not present an obstacle, and so he would be capable of grasping the content of even the most complex program.

With that, we do not mean to claim that the Laplacean demon would *immediately* gain *insight* or *understanding* by executing the algorithm by hand (say). However, it is usually far easier to understand what an expression says or means when one is aware of what it 'does'.

For instance, take a differential equation (DEQ), such as that for the damped harmonic oscillator. It is possible to get some idea of what a system obeying this equation will do, but it is only when one has the solutions in hand that one sees, quite immediately, the three possible behaviors (underdamping, overdamping, critical damping). Similarly, the ability to scrutinize each and every step of an algorithm should equip the demon with an excellent basis for understanding its workings, given his immense computational power.

Invoking the demon in this way, we can see that even essential opacity is closely tied to the epistemic conditions of an actual agent, and thus, in a specific sense, not a *principled* inaccessibility. As a first step towards clarification, we may therefore distinguish between *contingent* kinds of opacity, which depends on the actual epistemic conditions of a given cognitive agent *X*, or a community thereof, and *principled* or *fundamental* ones, which would apply to any agent *X*, regardless of their cognitive facilities.

Is not obvious that indisputable instances of principled opacity exist at all. However, in Bohmian versions of quantum theory, for instance, the precise values of quantities such as the velocity of an elementary particle cannot be found out by any conceivable experiment, despite the fact that they exist according to the theory (cf. Dürr et al., 2012, p. 139). Hence there are at least *candidate* cases of principled opacity.

To capture the relevant differences more clearly, let us try and make things more precise by formalizing the relevant notions. Humphreys' definition of opacity *O* of process *p* for agent[66] *X* at time *t*, with *X* having nature $N = n$ at *t* and some piece of information i_p about *p* being epistemically relevant, *R*, to *X* but not known, *K*, by *X* at *t*, is *relational* and triadic:

$$O(p, X, t) \leftrightarrow \exists i_p \left(N(X,t) = n \rightarrow \neg K\left(X, i_p, t\right) \land R\left(i_p, X, t\right) \right) \tag{4}$$

[66] Humphreys intends his definition to include human as well as non-human agents. For the purposes of our paper we will only consider human agents.

The agent's nature doesn't do any real work here and could be omitted, but it seems nice to keep it in place, because it will do all the work in the subsequent definitions.

In the definition of *essential* opacity, Humphreys doesn't appeal to time anymore, so it is merely dyadic. Moreover, Humphreys assumes the impossibility for *X* of acquiring everything that is epistemically relevant. Since impossibility implies non-occurrence already in the second weakest modal system however (cf. Hughes and Cresswell, 1996, pp. 41–2), we take it that one may replace talk of impossibility by talk of factive unavailability at all times without buying into any strong modal commitments. A nice (metaphorical) way of putting it is hence that in essential opacity, O_e, time is 'marginalized for', qua being bounded by a universal quantifier:[67]

$$O_e(p, X) \leftrightarrow \forall t \exists i_p \left(N(X, t) = n \rightarrow \neg K\big(X, i_p, t\big) \wedge R\big(i_p, X, t\big) \right) \qquad (5)$$

Our definition of *fundamental* opacity, O_f, now goes a step further in 'marginalizing' for both *X* and *t*:

$$O_f(p) \leftrightarrow \forall X \forall t \exists i_p \left(N(X, t) = n \rightarrow \neg K\big(X, i_p, t\big) \wedge R\big(i_p, X, t\big) \right) \qquad (6)$$

In words: given any agent with any nature, at no time will the agent be able to obtain all relevant pieces of information about *p*. Only 'supernatural' beings could possibly overcome fundamental opacities, for if one was to overcome one's own nature, one would still end up having some different agent's nature, and hence still miss some piece of information i_p.

Notably, we have chosen to represent an agent's nature by a dynamical variable here, and this may certainly raise some controversy. For should an agent's very nature not be fixed once and for all? What could a sensible alternative formalization that respects this intuition look like? The work done by the mention of an agent *X*'s nature here is to point to certain facilities and restrictions that *X* faces. The intuition says that these restrictions/facilities should be somewhat strongly tied to *X* itself. An obvious alternative would hence be to formalize *X*'s nature by letting *X*'s biological and cognitive *state*, $S(X, t)$, range exclusively over a set $\bar{N}$ such that $\bar{N} := \{n | n$ is nomologically/metaphysically accessible to $X\}$:

$$O_e(p, X) \leftrightarrow \forall t \exists i_p \left(S(X, t) \in \bar{N} \rightarrow \neg K\big(X, i_p, t\big) \wedge R\big(i_p, X, t\big) \right) \qquad (7)$$

[67] Note that the right-pointing arrow stands for several different relations in this paper: here, it stands for material implication, in other cases it indicates limiting behavior, in yet other cases a mapping, and sometimes even a decay-chain. It should, however, be clear from context what is meant in each case.

However, accepting this definition of *X*'s nature would render the impossibility of knowing i_p somewhat trivial: Given the fact that quantification over time is supposed to track impossibility, the definition in (7) basically says that obtaining i_p is impossible for *X* if it is impossible for *X*. Moreover, locutions like 'it was unnatural for me but I did it anyways' suggest that one's nature does *not* connect too intimately to an *impossibility* of sorts. Neither does the fact that transhumanism is, for all we know, both a nomological and metaphysical possibility, and it would be an obvious way to overcome (and thereby change) one's nature. Hence we find definition (5) preferable, and the intuitions motivating (7) somewhat misguided.

Given this level of precision, what is the significance of our earlier appeal to the Laplacean demon? The Laplacean demon, to recall, is "an intelligence sufficiently vast [so that] for it, nothing would be uncertain and the future, as the past would be present to its eyes." (de Laplace, 1814, p. 4) Gillies (2000, p. 14), moreover, points out that the Laplacean demon, at least in Laplace's sense,[68] is just an extrapolation of a human agent's capabilities. Indeed, we find Laplace state that: "The human mind offers, in the perfection which it has been able to give to astronomy, a feeble idea of this intelligence." (de Laplace, 1814, p. 4)

Of course, Laplace was concerned only with deterministic theories, containing rather 'tame', solvable DEQs. Today, this is not feasible anymore for the vast majority of physical theories. However, we are here interested in CSs and ML, methods that can be defined in terms of *algorithms*. Once this level of algorithm has been reached (say by translating the approximate content of a physical model into computer code for simulation), it seems that Laplace's demon would have an easy time figuring out the algorithm's workings.

Given also the connection we have drawn between algorithmic transparency and understanding, algorithmic transparency is hence enough to render even essential opacity *contingent*, qua contingent on an agent's nature. The Laplacean demon, who would be capable of biting through every step of the algorithm, would likely be able to digest and comprehend the algorithm's workings, and so may serve as an ideal *point of reference* for a

[68] There are various precisifications of the notion of a Laplacean demon in the literature: Frigg et al. (2014, pp. 33-4), for instance, define the demon to have access to initial conditions, possess a *perfect* model, and be able to compute *arbitrarily* fast. For our purposes, a more modest demon suffices that could come to know the boundary conditions of a given algorithm, compute *vastly faster* than any human being, and provide a highly detailed *error analysis* regarding hardware and software induced artifacts or the like. This will cover insight into the details of any realistic, humanly written computer program.

'natural' being here. Only if a method was opaque to him as well could it count as *fundamentally* opaque in our sense.

2.1 Computer simulations: Opacity induced by complexity

So CSs and ML come out as contingently opaque (qua algorithmically transparent) on our account. However, are the contingencies that lead to them the same? Indeed, there seems to be a major difference, which relates to the *kind of code*, i.e., to its purpose and associated definitions.

To see the difference, let's briefly recall the key features of the ATLAS simulation infrastructure: An 'event generator', factored into hard scatter, parton shower, hadronization, decays, and possibly an underlying event, feeds simulated 'final state particles' into a detector simulation, which then models the interaction of these particles with elements of the ATLAS detector, as well as its geometry.

However, as we noted above, detector simulations will rely on a vast range of different models, coming from different parts of physics. In some cases, such a 'model' will be highly phenomenological, i.e., not based on fundamental physical principles, or even simply amount to a function with free parameters, fitted to measured data. Reasons are that detailed information of the microphysics is unavailable because of the material composition and the complexity of interactions, or even missing information about the exact composition and purity of the material and the distribution of type impurities therein.

Moreover, while the structure of the hard scatter is basically dictated by perturbative quantum chromodynamics (QCD), one has to make certain assumptions, such as the transferability of *factorization theorems* to various unproven cases, which imply the factorization of proton/proton cross sections into calculable, elementary ones and a remainder called patron distribution functions. This factorization assumption in turn implies the need for a choice of an arbitrary factorization scale, where the choice has less and less influence with the inclusion of higher terms in the perturbation series.

Similarly, one can use either the separation angle between two partons or their transverse momenta as an evolution variable in parton showers, and different choices have traditionally proven fruitful in the two standard general-purpose generators PYTHIA and Herwig. There is also the need to cut the evolution off at some point, as relevant quantities otherwise diverge and cannot be interpreted in terms of the (probabilistic) Markov evolution that is the basis of standard shower-algorithms. In hadronization, finally, the energies are too low to use perturbative QCD at all, and one has to resort to one out of a range of phenomenological models, with a number of additional free parameters, to bridge the gap.

In midst this jungle of choices of modeling assumptions, parameters, conditions etc., who can really tell what a given simulation set up at ATLAS is precisely going to do? What is the effect of choosing certain parameters? Certain models/parametrizations in detector simulations? Certain (versions of) event generators? Combinations between them? On top of all that, documentation is usually sloppy, and quite a large number of people will have contributed individual pieces to this huge amount of code; so it is not straightforward to read up on the details and impossible to ask a (small group of) developer(s) about the full code in a simulation chain.

The source of this apparent opacity seems to reside in the *complexity* of the CSs, as well as of the underlying problem set. Complexity is here to be understood, first and foremost, in the sense that there will be numerous variables, parameters, and relations between these on the level of physics modeling (which includes models of measuring instruments; see GEANT Collaboration (2016)), and even more so on the level of implementation (which may require additional parameters to be integrated by appeal to further relations). Furthermore, this complexity is amplified, in a way, by the fact that there is a strong division of labour, i.e., that different programmers will be in charge of implementing the physics variables in terms of parts of the total code.

It seem that ATLAS is just a paradigmatic example in this respect though: in many detailed CSs in science, the interplay of a vast number of components, the complexity of the problem set, and the associated division of labour, will make it very hard to determine the precise goings on during the simulation, if not impossible for non-Laplacean demons.

2.2 Deep learning: Opacity induced by the method

The nature of the opacity associated with ML and DL has so far scarcely been addressed in the philosophical literature,[69] a fact which is presently changing, as reflected by the contributions to this volume.

A brief footnote-remark can be found e.g. in Schembera (2017, p. 60):

> machine learning could change the whole picture. Nevertheless, in terms of the algorithms, there is still strong transparency: All the steps of a computer program are determined, but the opaqueness of the paths of the steps as well as of the results, is increased so that they become not traceable.

This remark underscores our earlier observation of ML's algorithmic transparency: The functions implemented in DNs or other learning algorithms are laid down beforehand, and so ML is in the same sense algorithmically transparent as are CSs. Nevertheless, ML may be *additionally* accompanied by a *special* kind of opacity, which Schembera locates in the "opaqueness of the paths".

Humphreys (2004, p. 149) similarly hints at a unique kind of opacity associated with DNs:

> Epistemic opacity [...] plays a role in arguments that connectionist models of cognitive processes cannot provide an explanation of why or how those processes achieve their goals. The problem arises from the lack of an explicit algorithm linking the initial inputs with the final outputs[...] at a level that correctly captures the structural aspects of the surface representation[...], together with the inscrutability of the hidden units that are initially trained.

Humphreys here of course has in mind Fodor and Pylyshyn's (1988) influential paper on connectionism and classical cognitive architecture (cf. his Fn. 22), and interesting issues arise in this connection indeed.[70] However, Humphreys blames the lack of an explicit algorithm at a *humanly understandable* or 'conceptual' level (cf. ibid.), whereas ML is algorithmically fully transparent at the level of programming, as we have seen. Similarly, Burrell (2016, p. 2) speaks of a "mismatch between mathematical optimization in high- dimensionality characteristic of

[69] Exceptions are, for instance, Kaminski (2014) or, more recently, Sullivan (2019).
[70] We will return to these only in section 3.5, when scrutinizing the use of 'representation' in the context of DL, and only to a limited extent.

machine learning and the demands of human-scale reasoning and styles of semantic interpretation”

Hence, given ‘demonic’ computational powers, it might well be possible to figure out the workings of a learning algorithm on the basis of their algorithmic transparency, as was argued to be the case with CSs. However, with human-scale capacities, diverse tricks may be necessary for gaining understanding.

Another way to put the problem is that there is no (human-scale) algorithm that renders the *flow of information* through the net transparent to the epistemic agent. Again, complexity is at least in part to blame for the occurrence of opacity in this sense: the DNs used by Baldi et al. (2014), for instance, included five hidden layers with 300 units each. Assuming that each node in one layer was connected with nonzero weight to each node of the successor layer, this makes $\mathcal{O}(10^{12})$ paths along which information can travel between hidden layers. This makes it hardly conceivable that a simple algorithm can be found that allows any human agent to track the flow of information through the net.

However, complexity *alone* does not seem to be the culprit here: In stochastic gradient descent, for instance, we equip a DN with, e.g., backpropagation and parameter-updating algorithms and then let it ‘learn’ itself, based on these algorithms. Hence we convey a partial- or *quasi-autonomy* on the system, by letting it perform itself the iterative tuning of parameters that ultimately leads to success in the given task. This is different in CSs: here the scientist herself will intervene over and over, to tune a simulation program better to a specific task.[71] If this kind of control is given up, it becomes opaque in which ways the network was able to achieve

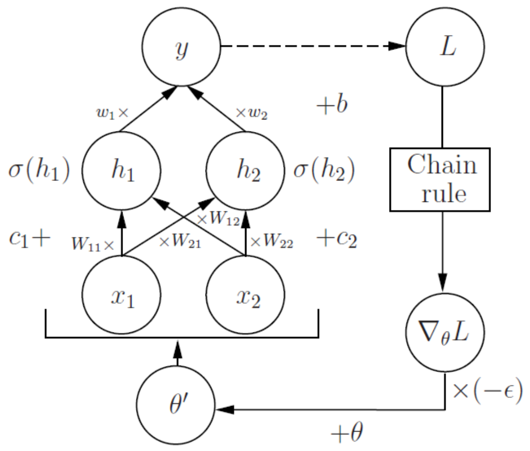

Figure 7: Pseudo-computational graph for stochastic gradient descent in a shallow network with sigmoid activations. (The backpropagation algorithm is not resolved in detail.)

[71] It may be remarked here that the same usually still applies to a network’s hyperparameters (cf. above).

the desired success. It is hence a kind of opacity *induced by the method* that is specific to 'learning' algorithms.

For concreteness, consider the simple, shallow network depicted in Figure 7, similar to the one from Sect. 1.2. In the network depicted here, the hidden units execute sigmoid functions, which makes their partial derivatives non-trivial. Figure 7 also displays an additional loop from the output back to the input, representing backpropagation together with stochastic gradient descent, which establishes the development of the network during the training phase. Assuming that the training is supervised, a loss function (denoted L in Figure 7) will be computed which compares the output to a target output t. For simplicity, assume also that the likelihood is Gaußian with variance 1 around t and the output binary. Then the loss function in formula (3) would, up to the regularizer term, yield $L = \frac{1}{2}(y - t)^2$, where *y* is a function of *b*, the w_i, $\sigma(h_i)$, c_i, and W_{ij}. Derivatives w.r.t. all parameters in θ will now have to be computed successively, which, by the chain rule, implies the need to compute derviatives such as $\frac{\partial L}{\partial y} = y - t$ or $\frac{\partial \sigma(h_i)}{\partial h_i} = \sigma(h_i)(1 - \sigma(h_i))$ and chain them up appropriately to compute the derivatives further down.

The goal of the training phase, to recall, is to minimize L, thereby effectively searching for a *y* s.t. *y* ≈ *t*. The stochastic gradient descent will now use the entirety of derivatives computed by backpropagation, plug them into a gradient ∇L,[72] and update the set of parameters θ by subtracting $\epsilon \nabla L$ from θ. This process will be iterated through a number of 'epochs' until a predefined proximity between *y* and *t* is reached. All this will be done without any intervention by the researcher, and therefore quasi-autonomously.

Despite the fact that more hidden layers can speed up the training, 10000s of epochs may be needed, producing a huge amount of individual steps that would need to be reviewed to track exactly how the final network is reached. Note also that we have here not only restricted our presentation to a shallow network, but also to supervised learning. In the case of unsupervised learning, where no target t is specified, it will likely be even more opaque how a network reaches its final configuration.[73]

[72] Since derivatives will be taken also w.r.t. matrix elements, one would, strictly speaking, first have to 'flatten everything out' into a column vector to use the procedure exactly as described here. This is just a matter of representation though.

[73] There are already applications of unsupervised learning in HEP. An example for the use of unsupervised learning in discrimination tasks is discussed, e.g., in Andreassen et al. (2018).

Supervised or not, it seems impossible to retrodict, from the final configuration, the steps taken by the network, and since individual steps are executed without direct guidance by the researcher, it is also impossible to simply check corresponding records. This justifies our claim that this is another kind of opacity, induced by the quasi-autonomy of the machine during the training phase, i.e. by the *method* itself. While complexity obviously plays a role, opacity is not purely due to complexity here.

However, this opacity induced by the method is still of a contingent nature: a Laplacean demon could compute all steps of the learning algorithm himself and, assuming knowledge of the inputs and initial weights and biases, thereby inspect all steps undergone by the network during the training epochs.

3 (Partially) overcoming opacity

Above, we have clarified three 'levels' of opacity ('ordinary', essential, and fundamental opacity) through careful analysis and formalization. Moreover, we used Laplace's (original) demon as an ideal point of reference for a 'natural' agent, which allowed us to group opacity and essential opacity into a class we coined 'contingent opacity'. Certainly, clarity is desirable, but what exactly is *interesting* about this analysis?

The fact is that the continuity between the demon and human agents suggests that humans could overcome contingent opacities *to a certain degree*: approximating the demon's capabilities by means of exploiting (second order) knowledge about the purposes and general properties of the code, as well as auxiliaries that one has more insight into, one may also gain some amount of insight into the workings of even a complex computer code. Without having a formal notion of degrees of opacity in mind, we shall now make it plausible that this could be achieved for ML methods in the future, and is in fact achieved, by some, to a remarkable extent for CSs in HEP.

3.1 Epistemic relevance, justification and understanding

The first step in this endeavor is to isolate the *epistemically relevant* elements that figured crucially in all three definitions. In this connection, Durán (2018, p. 108; emph. added) has recently pointed out that "researchers are only interested in a limited *amount* of information that counts for the *justification* of results." A detailed knowledge of the program code is presumably not relevant for justifying the results of a CS. Rather, validating well-designed code against known data and available benchmarks may justify some amount of trust in its results, insofar as success in these procedures is transferable. Hence justification of the *desired* knowledge, typically knowledge about some real-world target of the simulation, may convey relevance on some piece of information i_p about the simulation process p.[74]

Another hint for delineating epistemic relevance comes from Lenhard (2006), who relates epistemic opacity directly to *understanding*.[75] Several epistemologists (e.g. Brendel, 2013; Kvanvig, 2003) have indeed pointed out that we usually value knowledge not for itself but for its ability to increase understanding. Riggs (2003) and Pritchard (2014), moreover, connect understanding especially to *scientific* knowledge, which is certainly apt for most if not all CS-laden sciences.

Lenhard (2006, pp. 611–3), however, suggests to distinguish between an *insight view* and a *pragmatic view* of understanding. The insight view basically says that one understands a ‘device’, such as an equation, if one can determine properties of its solution, i.e., determine what it *does*, without necessarily being able to solve it / reproduce its behavior (cf. Feynman et al., 1965, Vol. 2, p. 2-1). The pragmatic view rather says that one understands the device if one is able to use it in order to exert control over phenomena.

Lenhard here bites the bullet and argues that CSs are accompanied by a loss of insight-like understanding regarding their own detailed functioning while promoting control over certain phenomena, thereby increasing our pragmatic understanding of these. In other words: a *method* that is opaque may render a certain *subject matter* more transparent: “Simulation offers an opportunity for ‘systematic inquiry’ that remains valid even under the condition of partial opacity.” (Lenhard, 2019, p. 122)

In a very similar vein, Kaminski and Schneider argue that CS can be an opaque way to overcome the opacity of certain mathematical equations. To argue for this, they first differentiate between strategies of justification by borrowing from a distinction in the theory of knowledge: “*Internalist* strategies of justification are based on *insight* into the interrelation

[74] It may be remarked here that Plato famously already highlighted this systematicity induced by justification, i.e. the ability to ‘stabilize’ knowledge by other knowledge through justification, as a central value of knowledge over mere true belief in the *Menon* (97A–98A).

[75] 23As Andreas Kaminski has kindly reminded us, this connection is also present already in Humphreys’ original approach (see, for instance, Humphreys, 2004, Sect. 5.3).

of elements. The prototype of this is the argument and the mathematical proof. *Externalist* justification strategies are based on the *reliability* of a process." (this volume; first and third emphasis added, second and fourth in original)

The appeal to this distinction in the context of CSs and their opacity is that CSs can serve as external means of justification for certain mathematical results while remaining themselves entirely *opaque*. We will address this issue later. However, note here already that the position that justification does not immediately imply an increase in understanding (and thus a reduction of opacity) is not entirely unrivalled: Durán (2018, p. 106) buys into the same premise but rejects the conclusion: "Reliabilism[...] is the most successful way to circumvent epistemic opacity."

As shall become clearer a little below, in our view, insight-like understanding *is* required to render a CS (more) transparent. Hence, if we want to argue that CSs in HEP are, or can be made, transparent to a respectable degree, we also need to show how physicists gain insight into their workings, not just how they can use them successfully.

Note that the Feynman-approach to insight does *not* require the ability to *solve* the equation / *walk through* the algorithm for claiming insight: It is exactly the ability to determine what either does *without* having to solve / go through it that is crucial for being able to claim insight into it (cf. similarly Kaminski et al., 2018, p. 260, for the difference between execution and understanding). In a similar spirit, Durán (2018, p. 108) has recently pointed out that a detailed understanding of each and every line is certainly not required to understand what some code *does*. Rather, an assessment of the way in which it *functions* as well as of the inter-relations between its different parts is in order for such a purpose.

In sum, we take it that any piece of information i_p about process *p* that connects some 'input' *s* with an 'output' *y*, where an epistemic agent *X* is interested in *y*, will only be epistemically relevant to *X* if i_p either (a) conveys justification on accepting the output as serving an intended epistemic purpose (such as adequately representing the states of a physical system) or (b) serves to convey an insight into the way in which *p* leads from *s* to *y*.

If all this information was available, *p* would be perfectly transparent. As we have shown, however, the right amount and level of information to seek for justification or insight respectively does not necessarily reside in the details of the code in a CS or ML technique.

This gives us an edge in approximating the demon: we do not have to acquire his immense computational capacities to get deeper insights into as CS or ML algorithm; clever shortcuts can promote understanding while leaving certain (epistemically irrelevant) details in the dark. For recall that the demon's capabilities were identified only as an ideal point of reference, in which a particularly helpful means (immense computational power) for acquiring transparency is present (cf. the example of the damped oscillator DEQ and its solutions)—*not* a necessary condition.

3.2 Paths to overcoming opacity

Given these preliminaries on epistemic relevance, how could agents 'overcome their natures' to render essentially opaque processes more transparent? The most obvious answer would be transhumanism, i.e. the integration, say, of suitable equipment into one's brain, or the 'upload' of the mind into a supercomputer. But these are presently just science fiction fantasies, so we'll leave it at that.

More realistically, agents may *team up*, divide the labor, pack the results of individual efforts into a more accessible form, and then achieve transparency collectively. By this we do not mean that the collective as a *whole* achieves transparency, but rather, that through collaboration each *individual* from a skilled group with a good division of labor may achieve a good deal of transparency.

The former option (group transparency) would be compatible with Durán's appeal to reliabilism: nobody would really 'have' the justification, but it could still claimed that (trust in) the CS 'is' justified. However, we are skeptical about reliabilism as a general account of knowledge, let alone understanding. Given what was said about the individuals in dedicated collaborations, out position is closer to the one coined 'reliability internalism' by Schurz (2008, p. 21; orig. emph.): "a justification of *X*'s belief-that-*p* is a *system of arguments* which *demonstrates* that the mechanisms which caused *X*'s belief-in-*p* are reliable, thereby recurring to *conditions* which are *epistemically accessible* for human beings."

The point is that "justification [...] must be cognitively *possessed* [...] by some *competent attributor* [...]." However, to be a competent attributor of justification based on simulation, one has to understand a fair deal about the simulation, as well as what can be achieved by validation, benchmarking, etc. Competent attributors could be present in the sense of a weak ordering of quantifiers; i.e., for every piece of information (insofar as available at all) relevant to the justification of some simulation result, there is a competent attributor (an expert sufficiently skilled in scientific communication, at the very least for the purposes of the group of experts) who has that information.

When taken together, the information had by the individuals in such collaborations may serve as the relevant 'system of arguments' for justification. However, the need of a scientific agent who is in possession of at least part of that system underscores our earlier claim that we think insight-like understanding is required for (increased) transparency. A CS for which this situation obtains may be called (largely) 'third-person internally justified', following the terminology in Schurz (2008).

Alternatively, an agent could also exploit auxiliary *devices* or techniques whose functioning is sufficiently transparent to her. It is a case-by-case question whether any of these techniques work – which Humphreys (2009, p. 619), thinks, isn't fulfilled for most CSs. But below we will make a case that this (especially the possibility of collective third-person internal justification) is realistic at least for large chunks of CSs utilized by the ATLAS collaboration.

Emphatically, we do not mean to claim here that, in fact, each and every physicist at ATLAS, say, fully understands the simulations she uses. As already emphasized earlier, we do not even believe that there is even one individual at CERN who has full insight into the workings of all the complex CSs used for the LHC. However, we *do* believe that in dedicated collaborations, individual scientists can become able to obtain a *fair deal* of insight into, say, a particular version of the Pythia event generator or the GEANT detector simulation toolkit; simulation programs that have been in use for a long time and extensively studied.

Next to what we have learned from practitioners (in private communication), this verdict concurs, e.g., with that of Zach Marshall on the ATLAS blog: ATLAS "wouldn't run without the simulation, and yet there are few people who really understand it." (see https://atlas.cern/updates/atlas-blog/defending-your-life-part-1; checked 08/20) This quotation implies that the number of people who consider (parts of) the ATLAS simulation infrastructure rather transparent is small—but *non-zero*.

Different verdicts have been drawn, however, by Kaminski et al. (2018), as well as Kaminski and Schneider (this volume), who have recently identified several sources, or aspects, of opacity that they consider relevant for CSs, namely *social*, *technological*, and *mathematical* ones. The social aspects reside in the fact that one usually builds on the works of others in science; in a CS, e.g., on libraries or modules that one has not programmed oneself, whereby one typically lacks insight into them.

The technological ones are to be understood as a want of deeper understanding of any sort of technical equipment or, more generally, technique that one uses. In a CS, this comprises the computer hardware, but also established numerical (or even coding) techniques: "Without having to know by whom a mathematical technique was discovered and why it 'works' (i.e. without knowing the proof), it can be used as a tool (provided the prerequisites are given)." (Kaminski and Schneider, this volume) The mathematical reasons, finally, reside in the fact that, for many mathematical equations as used in science – and especially as the basis for CS – it is unknown how solutions to these would behave, and whether they include properties such as strange attractors or bifurcations.

For Kaminski et al. (2018), the possibility of collective or technology-aided transparency we have sketched above would be doubt worthy, and in a sense we agree: auxiliary devices and other agents may introduce another layer of sources of opacity. However, depending on the communication skills and technological abilities of individual agents, the opaque remainder may be reduced, and at least for certain aspects or parts of a complex simulation, even become close to negligible, depending on one's standards for epistemic *relevance*. As we have pointed out above, this may not be so far from the truth in actual examples.

3.3 Complexity: Learning about model relations

Let's now first consider the overcoming of merely complexity-induced opacity, as it attaches to the simulation infrastructure at ATLAS. Indeed, we already admitted that documentation may often be sloppy, which can make it hard to comprehend. Nevertheless, a tremendous *amount* of documentation exists: The $\sim$ 250 000 lines of code in the Athena framework specific to ATLAS simulations, for instance, are accompanied by some $\sim$ 70 000 lines of commentary (ATLAS, 2010), as well as various user manuals and wikis providing overall details on the CSs' functioning to the user (e.g. https://twiki.cern.ch/twiki/bin/view/Main/IUedaAtlasSimulation#Atlas_Simulation or ATLAS Computing Group 2005, for examples). It seems that getting insight at least into the detailed workings of pieces of written code is largely a matter of *personal effort*, not one of principled restrictions.[76]

In particular, while the code-writing is, indeed, often sloppy, there are organized efforts for cleaning up codes and for streamlining future code development, such as the High Energy Physics Forum for Computational Excellence (Habib et al., 2015). On top of that, communication between experimenters and dedicated software-development groups appears to function rather well, so a lively exchange exists between the needs of experiments and the writing of code for these very purposes (cf. ibid.).

However, insight also requires knowledge of the *interactions* between parts, and we had also identified the justification of results as a second key condition for epistemic relevance. The Athena code will be constantly validated against benchmarks, which secures that it works as intended *as a whole*, even when new code is added. This conveys some amount of justification.

More importantly, though, there is also dedicated small- and large-scale *integration testing*, meaning that, when pieces of code are replaced, their joint functioning with the remainder of existing code is tested for on small and large scales (cf. ATLAS Computing Group, 2005, p. 78 ff.). In other words: when a new piece of code has been written, it will be immediately checked whether, and in what ways, this alters the functionality of the remaining code.

Three frameworks are used for this, called AtNight, Kit Validation, and Run Time Tester (RTT). There are differences in how these are used: RTT, for instance, "is currently run daily on a farm at UCL", and the "full set of RTT results currently take a number of hours to generate." (ibid., p. 79) AtNight, on the other hand, is used to test the 'nightly builds', i.e., automatically generated parts of the total code that are still at an early stage of development. Developers have a number of choices in these frameworks as to what to test and how (cf. ibid.).

These tools all fulfill somewhat complementary roles. For instance, while AtNight is mostly concerned with software-specific properties, including the interfacing with other modules of the overall software, RTT "sets up relatively complex jobs [...], runs them, performs post-

[76] Andreas Kaminski has pointed out to us that this does not address the interaction with the compiler yet. We agree, but given the above considerations on epistemic relevance, we are not convinced that these specifically implementation-related aspects cannot be compensated in large part by suitable testing, including and especially *integration testing* (see below).

execution actions and tests and finally publishes the results on a web accessible location [...].” (Costanzo et al., 2010, p. 2) Due to its computational expensiveness, this is done only for some tens of events. However, subsequently, a(n even smaller) number of full-chain test is run, ranging “from basic Standard Model processes [...], to exotic (and very complex) events (black holes production).” (ibid., p. 3) It would be quite surprising if substantial errors would remain undisclosed by these diverse means of addressing the functionality of the code. Moreover, because the ‘statistics is high’ in HEP (there is loads of data), divergences can be identified quite precisely. The rate of errors induced by bugs in the software is reported to lie in the order of some $10^{-4} - 10^{-5}$; numbers deemed “an acceptable failure rate” by working software developers in HEP (ibid., p. 5).

That by itself would, however, only establish *reliability*, not necessarily (increased) *transparency*. Given the importance of insight-like understanding for human agents that we recognized earlier, it is thus vital to realize that the code has been written *purposefully*, *by physicists*, and that each part of it will derive from some sort of physics model (cf. Boge and Zeitnitz, 2020, for some details).

These models either rest on a solid theoretical basis or relatively simple assumptions, and so may be claimed to be as well-understood as any physics model: At least for high energies and to the extent that it is even possible to ‘understand’ a quantum theory,[77] quantum chromodynamics is rather well-understood, in terms of a perturbative treatment. This means that Feynman diagrams computed to fixed orders in (powers of) the strong coupling constant deliver sensible results that can be mapped to experiments, and the higher order terms left behind can be expected to have smaller and smaller impact on these results.

At lower energies, this ability to compute and comprehend the theory breaks down. But there, simplifying, idealizing models take over. The assumptions of these models, such as the string model of hadronization, are typically rather ‘domestic’. For instance, in string models of hadronization, the ‘tearing of the string’ that represents the breakup of the color field that binds quarks, which in turn corresponds to the creation of two new mesons, is predicted by a simple probability density whose shape is derived fully from physical and simplifying modeling assumptions (Andersson et al., 1983). No skilled physicists will have a hard time

[77] de Regt’s (2017) account of understanding, for instance, would allow this.

comprehending its origin and properties. At the level of physics modeling, there is, in other words, no reason to expect a special sort (or even amount) of opacity.

The *relations* between these models, moreover, can also be understood in detail before programming, and dependencies between model parameters will be known and respected by job specifications allowed by Athena, or even already compensated for in the code.

For a detailed discussion of these relations we refer the reader to Boge and Zeitnitz (2020). As an example of compensation of dependencies, however, consider what is called 'matrix element matching': In the simulation of an event and subsequent processes, "double counting can arise when the same final state can be generated both by the parton level calculation, and by the showering algorithm." (Mangano and Stelzer, 2005, p. 564) The details are intricate (cf. ibid., pp. 579–80) and shall not be recaptured here. In essence, however, "a probabilistic rejection procedure [is] applied to events falling in the overlap" between phase space sectors associated with parton showers and hard scatters respectively, "to ensure that a given phase-space configuration is only counted once." (ibid., p. 579)

Hence already on the level of physics modeling (with the goal of simulation in mind), will insight be gained by studying relations between different models and modeling steps in detail. Moreover, several pure programming artifacts are known and compensated for as well: an example are 'stuck tracks' in the detector simulation, due to an infinitely dense spacing of the simulated components.

The above suggests that the study of model relations pre-implementation, together with integration testing after the code writing, secures some amount of insight into the code: If what is already understood on the physics-level is preserved in the computer code, this means that most of the already obtained understanding carries over.

However, transparency is thus gained also on the side of justification; specifically, in the ability to attribute *errors*. Consider the following quote from Boge and Zeitnitz (2020, p. 30):

> Testing the integration-properties of a certain piece of code—after having taken into account the connections that already exist across the underlying physics models [...]—means checking whether including this code (or the model it represents) will spoil the validity established for the integrated whole. When this is the case, it is possible to

attribute at least the *failure* to the new piece of code or to its connection to other pieces, and to re-assess it on the physics and implementation level.

In this way, the detailed integration testing performed by ATLAS (or, more generally: LHC) physicists at least in part compensates the following observation by Kaminski and Schneider (this volume; orig. emph.): “Not all errors must be relevant, but *it is not known apriori which errors are relevant, and which are not*.” The interplay between overall validation and small- and large scale integration testing allows to locate errors *known* to impair overall functioning in certain pieces of code. Hence, if there are still errors contained in the code (as there always are), or even related to hardware limitations and the like, these can be hoped to be irrelevant insofar as they do not seem to (vastly) alter already secured results.

This does not address results as yet to be secured, however. In this respect, considerations of *robustness* are helpful for sorting out artifactual results from those that can be considered trustworthy. We cannot go into much detail here (Boge, ms, for a deeper look); but roughly, the idea is that convergent results arrived at on the basis of very different idealizing and simplifying modeling assumptions and implementations (as available and used in HEP) can be considered trustworthy, i.e., probably *not* the result of particular programming and modeling artifacts, but rather consequences of a solid, common core.

We conclude from this that, at least for CSs used at ATLAS, “acceptability” is not *only* judged “according to the model’s *overall* behavior,” as Lenhard (2006, p. 613; orig. emph.) has it, but also by the components’ behaviors and by their interrelations. In other words: pace Lenhard (2006) and Humphreys (2004, p. 148), high energy physicists *do* obtain at least a fair amount of the “traditional transparency” connected to an “understanding[...] based upon the ability to decompose the process between model inputs and outputs into modular steps, each of which is methodologically acceptable both individually and in combination with the others.”

In sum: Basing epistemic relevance in justification and insight-like understanding renders many details of the code for ATLAS CSs epistemically irrelevant. Accordingly LHC physicists in dedicated collabo- rations may be said to achieve a fair deal of transparency by means of careful modeling, programming, documenting, and testing.

One may wonder why this level of transparency appears to be reached in HEP, but maybe not so much in other simulation-heavy fields like climate science. Three reasons offer themselves in particular: (a) In HEP one relies on proper experiments, which implies a high degree of control over the objects of study (e.g. Radder, 2009; Schurz, 2014), whereas in climate science one is largely limited to field observations. (b) There is some solid theoretical core in HEP (the Standard Model), which can be used to guide the development of CSs, whereas in climate science there is a plethora of models that start from rather diverse assumptions (e.g. Parker, 2018, Sect. 4.1). And, maybe most importantly, (c) HEP experiments have a high degree of social organization, which allows to monitor and review, among other things, software developments much more closely than in disparate disciplines like climate science.[78]

It is clear that several aspects, including certain sources of error, will remain opaque nonetheless. What we have shown, however, is that, depending on one's stance on epistemic relevance, HEP simulations that haven long been in use can be viewed as transparent *to a respectable degree*, for several individuals in dedicated collaborations. Since this is the result of collaborative effort, good communication, and the skillful application of techniques for obtaining knowledge about whether the implemented algorithm can be expected to perform as intended (and where it will fail), it follows exactly from the set of 'tricks' we outlined above for overcoming one's nature and approximating the demon.

3.4 Method: Learning about the learning process

Now what about the acclaimed 'black-box' nature of ML techniques? Worries of this kind may be connected to some general facts about ML, in particular that certain DNs have been known to suffer from *robustness issues*, in the sense that learned strategies will fail under even minute (though often specific) differences in inputs (cf. Marcus, 2018, p. 8). This first and foremost indicates difficulties in the *justification* of ML *results*, however, and may be compensated for by empirically testing for robustness in specific tasks (which is generally claimed to be rather high in HEP; cf. Guest et al. 2018), determining, as far as possible, the source of the non-robustnes otherwise, and improving on algorithms for individual tasks.

Indeed, there is the so called "no free lunch theorem", which "states that, averaged over all possible data-generating distributions, every classification algorithm has the same error rate when classifying previously unobserved points." (Goodfellow et al., 2016, p. 111) The implication is

[78] For a more detailed treatment of these issues, see also Boge and Zeitnitz (2020, esp. Sect. 4.3 and 4.4).

> that we must design our machine learning algorithms to perform well on a specific task. We do so by building a set of preferences into the learning algorithm. When these preferences are aligned with the learning problems that we ask the algorithm to solve, it performs better. (ibid.)

These preferences will be implemented in terms of regularizers $\lambda\Omega(\theta)$[79] added to the loss function. An example is the 'weight decay' regularizer λx^2, where $\lambda > 0$ will force the weights to become smaller, as $\nabla w^2 = 2w$, which will approximate zero only in case $w \rightarrow 0$.

We have argued, however, that the network's ability to self-correct introduces an opacity different in kind from the merely complexity-based one. If opacity is hence connected to this partial autonomy of the network, it may be difficult to determine the source of non-robustness.

Moreover, empirical steps for securing that some DN or boosted decision tree achieves the desired goal, i.e., justifying its results, would do no good for overcoming opacity in the sense of conveying *insight* into learning processes. The case is asymmetric to that of CSs in that we implement a certain *physics model* in a CS, and by performing benchmarks and integration testing we get an insight into whether this model (which, to recall, we argued to be generally well-understood in HEP) was implemented *as intended*.

Emphatically, by that we do not mean that empirical success *alone* produces understanding. Rather, against the background of the understanding gained by investigating the physics relations contained in / persisting between the models, and putting in effort to preserve these as closely as possible in programming, the empirical evidence gained through validation and integration testing sheds light on whether this process was successful. In that way, it *confers* the understanding had on the physics level on the implemented algorithm.

This is different in ML: Here the algorithm is basically just a method for extracting information from data in a largely model-independent fashion,[80] and for the reasons named above, the insight we may desire is exactly as to *how* this information was extracted.

[79] Thomas von Clarmann (private comm., 2019) has pointed out to us that understanding the exact functioning of regularizers often poses additional challenges. This implies that rendering algorithms that include regularizers transparent may require additional effort. It does not impair (but rather underscores) our general point though: that there are conceivable techniques for rendering ML largely transparent, in individual cases.
[80] Cf. also Humphreys (2013) on this point.

How do ML scientists cope for this kind of opacity? Indeed, protocols here are not as far developed as is the case with CSs at ATLAS, and the quest for transparency, usually subsumed under the header 'explainable artificial intelligence' (XAI) is more of an open field of research (e.g. Russell et al., 2017, p. 94). Yet some remarkable approaches and results already exist that sketch a path to XAI and allow to render specific ML techniques more and more transparent (e.g. Ronen, 2017, for an overview).

To give a concrete example, let us briefly zoom in on the approach by Schwartz-Ziv and Tishby (2017), which used information-theoretic quantities to track the evolution of a DN with stochastic gradient descent. A central insight underlying this approach was that a DN without skip connections (cf. Sect 1) can be seen as a *Markov chain*: each layer corresponds to a random vector whose distribution will depend only on its direct predecessor. Using the so called *information plane* (cf. Schwartz-Ziv and Tishby, 2017, p. 5), the authors were then able to track the evolution of DNs with 1–7 hidden layers of decreasing size (2–12 nodes) towards the output. The information plane is defined by considering the distributions $p\left(h^{(n)}\middle|x\right)$ and $p\left(y\middle|h^{(n)}\right)$ for hidden layer $h^{(n)}$ and using the *mutual information* $I(h^{(n)};x)$ and $I(y;h^{(n)})$ respectively as *x* and *y* coordinates of a Cartesian coordinate system. Here,

$$I(v;w) \coloneqq \textstyle\sum_{i,j} p\left(v_i,w_j\right) log_2 \left(\frac{p\left(v_i,w_j\right)}{p(v_i)p\left(w_j\right)}\right), \tag{8}$$

can be interpreted as the average number of bits of information *v* contains about *w* and vice versa.

Now Schwartz-Ziv and Tishby (2017) performed a bunch of numerical experiments in which for configurations of 12 uniformly distributed points on a sphere a yes/no decision had to be made, yielding 2^{12} = 4096 possible input configurations. To generate a joint distribution *p* (*s, y*), a *logistic sigmoid*,

$$\sigma(\,f(x)-\theta) = (1\,+\,exp[-\gamma(f(x)-\theta)])^{-1}, \tag{9}$$

of some spherically symmetric function *f* (x) of the pattern (uniquely indicating the pattern itself) was computed and interpreted as the probability $p(y=1|x)$. This can be understood by realizing that $\sigma(f(x)-\theta)$ approaches the Heaviside distribution $\Theta(f(x)-\theta)$ for growing

γ,[81] which corresponds to a deterministic yes/no decision in which $f(x)$ is compared to a criterion $\theta(f(x) \geq \theta$ means yes, everything else no. The sigmoid hence created a probabilistic rule for a yes/no decision conditional on *x*, which would yield a joint distribution $p(x, y) = p(y = 1|x)p(x)$, with $p(x)$ chosen uniform as indicated above. θ was chosen such that $p(y = 1) = \sum_x p\,(y = 1|x)p(x) \approx 0.5$ and γ large enough such that the mutual information $I(x; y)$ as approximately 0.99 bits (the decision close to deterministic).

The network's nodes executed an arctan activation function, which was binned into 30 equal intervals between −1 and 1. This binning was used to compute an approximation for the joint distributions $p(h^{(n)}, x)$, from which all other quantities, in particular the information plane-coordinates, could be calculated (cf. Schwartz-Ziv and Tishby, 2017, p. 8). This was repeated for 50 randomly chosen initializations of the network (weights and biases), as well as configurations *s* (ibid.).

One of the central insights from these experiments was the existence of two distinct phases in stochastic gradient descent:

> In the first and shorter phase the layers increase the information on the labels (fitting), while in the second and much longer phase the layer reduce the information on the input (compression phase). Schwartz-Ziv and Tishby (2017, p. 3)

This is to be understood in the sense that the layers first moved up along the $I(y; h^{(n)})$-axis, thereby increasing the amount of information each layer would provide for *y*, and then move to towards zero (towards the left) on the $I(h^{(n)}; x)$-axis, thereby *reducing* the information each layer contained about x. This may be seen exactly as a fit-then-optimize learning strategy: The networks first learned to achieve the goal (produce the desired *y* from given *s*) and then removes all the information about the input not required to achieve that goal with the same accuracy.

Moreover, by minimizing the quantity $I(h^{(n)}; x) - \beta I(y; h^{(n)})$-independently over all $p(y_i\,|h_i^{(n)}), p(h_i^{(n)})$ and $p\left(h_i^{(n)}\middle|x_i\right)$, one finds a set of equations, parametrized by β, that

[81] To understand this, consult the graphs of these functions and consider γ a measure for inverse width in the area where the curve has a steep (respectively: vertical) incline.

define a curve through the information plane on which $h^{(n)}$ may be said to be an efficient representation of *x*.[82] Ultimately, the networks approximated points on that curve.

A second result of the study, which could be explained with the help of the foregoing one (cf. Schwartz-Ziv and Tishby, 2017, p. 12), is that more layers will speed up the training phase. This may be understood on the basis of the optimization (removal of unnecessary detail) performed by the earlier layers, which provides already suitably compressed information to the later layers and makes their convergence to a desired result easier.

This is an impressive insight into stochastic gradient descent in a specific feedforward network, but of course not a general result about all DL, let alone ML. However, mutual information is invariant under invertible transformations of the variables; so the general approach can be carried over to a whole range of DNs. Moreover, the task was repeated with non-symmetric distribution over the sphere, and no difference in the general results was found. This also suggests a carry-over to a range of similar classification tasks. Finally, this is just one exemplary study in a broader research field, as we have pointed out above. Other studies may shed light on other kinds of learning machines or provide complementary insights on stochastic gradient descent.

Obviously, the procedures explored here are rather different from what is being done to overcome the complexity-induced opacity of involved CSs at ATLAS. This underscores our earlier claim that the opacity in question, and accordingly the information required for achieving transparency, is different in kind. However, on a high level of abstraction, quite similar reasons are relevant to the establishment of transparency: a learning algorithm is *purposefully* designed, implying already a fair deal of knowledge about its constitution. Testing corresponding DNs on a broad range of applications can then delimit the scope of the algorithm's suitability and compensate for lacking a priori knowledge during the design stage. This will covey *justification* on the acceptance of a range of results, to the extent that the tasks resemble each other in relevant respects. Moreover, using methods like that of Schwartz-Ziv and Tishby (2017), one can obtain a fair deal of *understanding* of the process the machine

[82] This is possible only if the output layer is not a deterministic function of the input layer. However, any deterministic function can be made stochastic by assuming that there is always some noise, and by correspondingly adding some minor stochastic error. This can be implemented exactly in terms of (probabilistically interpreted) sigmoid functions with their width representing the amount of noise (cf. Schwartz-Ziv and Tishby, 2017, p. 6 and below).

undergoes during the training stage. All in all, this may compensate for a considerable amount of opacity induced by the machine's relative autonomy. When all this is done correctly, the missing epistemically *relevant* information may become almost negligible here as well. But at present, XAI appears to be still pretty much in its infancy, and there is hence a de facto asymmetry to CSs at ATLAS.

3.5 Model-opacity: What do we really learn?

We have thus sketched an exemplary way in which the learning process of a machine may be rendered more transparent, and the opacity induced by the method may be overcome to a considerable degree. However, physicists such as Chang et al. (2018), from which we took the 'black box'-quote, are interested rather in the *physical features* on the basis of which a given DN learns to classify, not exactly in the way in which it does so.

There are, in other words, two *dimensions* to the 'black box'-ness of ML and DL methods, namely *how* the machine learns and *what* it learns. The approach by Schwartz-Ziv and Tishby (2017) discussed above targets only the first dimension. Chang et al. (2018), in contrast, tried to understand what physical features would yield the discriminating power for a DN. To that end, they used what they called a *planing scheme*, which essentially means removing information regarding certain variables in a data set and observing consecutive changes in the DN's discrimination power. The 'planing' refers, more precisely, to the weighting of events by a quantity inversely proportional to the differential cross section $\mathrm{d}\sigma(x_i)/\mathrm{d}m\,|_{m=m_i}$, or a relevant distribution more generally, where x_i labels an input vector for event *i*, and m_i means the value *m* takes on in that event.

In practice, Chang et al. (2018) inverted histograms, meaning that events would be weighted by 1/(#counts in bin *i*) for bins of the histogram corresponding to variable m, making this histogram itself a uniform distribution. Performance drops were then measured in terms of a reduced AUC, for an extension of the SM in which a new (vector) boson *Z′* was introduced that decays into an e^+e^- (i.e., electron-positron) pair and couples to SM particles. Two cases were considered: Z'_V where the couplings to right-handed[83] and left-handed SM particles would be

[83] Recall that the Dirac equation is solved by a spinor-field with four entries. For massless particles, the upper part then corresponds to negative eigenstates of a helicity operator ('right handed ones'), meaning that their spin lies in the opposite direction as their momentum, whereas for the lower part the two are aligned ('left

identical, and $Z'_L$where the boson would couple exclusively to left-handed SM particles. Data in both cases were produced using a Monte Carlo simulation, with 10^6 proton-proton collisions. The relevant background comprised standard pair creation events $\gamma^* \to e^+e^-$, with γ^* a(n off-shell) photon.[84] The mass for both Z's was assumed to be $M_{Z'}$= 1 TeV and the characteristic expected lifetime such that the corresponding peak at 1 TeV in the mass distribution would have characteristic width $\Gamma_{Z'}$ = 10 GeV. The photon background, in contrast, would simply be decreasing.

Two variables were then planed for in order to investigate performance drops: the reconstructed invariant mass *m* of the final-state products, as in the benchmark case discussed above, and the rapidity $y = \frac{1}{2}\ln[(E + p_z)/(E - p_z)]$, or rather the difference $\Delta y \equiv |y\,(e^-)| - |y(e^+)|$, which provides information about the angular distribution and hence indirectly about the asymmetry between couplings to right and left-handed SM particles. Actually, in the case Z'_V, only *m* was planed for, because of the (symmetric) way the example was set up. The AUC then approached 0.5 upon planing for *m* in this case, which means[85] that "no noticeable discriminating power remains." (Chang et al., 2018, p. 4) This was also compared to the performance of a linear network (no hidden layer), which would, as expected, perform significantly worse than the DN before planing when only fed with four-momentum information, but almost as good when equipped with joint information on *p* and m.

In the Z'_L case, on the other hand, some discriminating power was left in both networks when only *m* was planed (AUC>0.6) and a high correlation (0.90) could be established in the linear responses (before activation with a sigmoid function by the hidden units) with the variable Δy. Planing for Δy as well, the performance of both networks then approached 0.5 again.

Bottom line: planing selectively for certain variables, one gets an indirect insight into the physical char- acteristics (angular anisotropy of the distribution, therefore unequal couplings

handed' ones; e.g. Griffiths, 2008, pp. 338 ff., for an overview). The names are common for massive particles as well, although the identification with helicity eigenstates is not possible here.

[84] 'Off shell' is a term that refers to the shape of the (shell-like) boundary for the volumes in space-time diagrams within which the curves traced out by massive particles can lie. Being off this shell means that these particles defy the relativistic energy-momentum relation, which is possible only for 'virtual' particles, i.e., that (if anything) whichever is represented by the internal lines of Feynman-diagrams, which explicitly contrasts with the external lines that represent measurable particles.

[85] The AUC is standardly normalized to unity, so an area of 0.5 means that the best one can get is a joint 50% chance for correct positives and negatives. This in turn essentially says that the classifier is doing guess work instead of systematically classifying events.

to left- and right-handed particles; high information content in mass-distribution, therefore massive new particle) underlying the data. But the model was already known here, and the variables to be planed for were hence suggested on the basis of that model. So even if such a "procedure can be explored systematically", i.e., by planing iteratively for all conceivable physically relevant variables, it is "most efficient in tandem with physics intuition." (Chang et al., 2018, p. 5)

We must stress, at this point, that *one does not retrieve a model from the procedure, and a lot of variation in detail remains possible.* Let us refer to this fact as *model-opacity*. It is the opacity of the underlying physics, which could be captured by a mathematical model and is responsible for the machine's ability to discriminate, i.e., leads to the phenomena made accessible via the machine. It seems that there is no algorithm which leads us from the machine's output to a concise understanding of the underlying physical mechanisms. So even a Laplacean demon might be lost, as, in contrast to the kinds of opacity discussed above, we seem to have come across a variant that is *not* algorithmically transparent.

The problem set associated with this kind of opacity is acknowledged in the ML community typically under the header of 'causality':

> what makes one representation better than another? One answer[...] is that an ideal representation is one that disentangles the underlying causal factors of variation that generated the data, especially those factors that are relevant to our applications. (Goodfellow et al., 2016, p. 544)

In fact, supervised learning and a *distributed representation* may be utilized exactly with the aim of providing the machine with a means for learning the relevant factors of variation (ibid., pp. 544–5). However, we indicated above that one should be cautious about the meaning of 'representation' here. Consider, for instance, the following observation by van Fraassen (2008, p. 7, emph. added): "A representation is made with a *purpose* or *goal in mind*, governed by criteria of adequacy pertaining to that goal, which guide its means, medium, and selectivity." Similarly Suárez (2003, p. 237; emph. added): "*A* [...] cannot stand in [in a representing relation to *B*] unless it is *intended as* a representation of *B* by some suitably competent and informed *inquirer*." There is, in other words, a good case that representation requires intentionality, which in turn may require conscious content such as agreed-upon standards. In what sense, then, may a network be said to represent a given function, such as a likelihood ratio? We take it that it can represent such a function *to us*, in the sense that it does everything that the likelihood ratio does: it spits out a number of 'events' in the right frequency relative to expected 'background events'.

Now given that a likelihood is of the form *p*(data|hypothesis), there is a problem associated with the nature of the input here. For when we train a network on the basis of Monte Carlo data, resulting from the implementation of some approximation to a physical model, we know exactly what the 'hypothesis' is on which we condition and which gives us the input to be classified: it is a part or consequence of the implemented model. When we let a well-trained network loose, in contrast, on a range of previously unexplored data (say in a new energy range achieved with some future collider) and it indicates the presence of a completely unexpected phenomenon to us, then the network really just represents some sort of distribution, not a well defined likelihood; or more colorfully: a likelihood conditioned on an *unknown* hypothesis.

In sum, talk of representation is misleading here in two ways: (a) the network itself will not *literally* represent a given function, in the sense in which a conscious scientific agent would if she had a mathematical representation of it. And (b), even if we can track what the network does and thereby work out the function 'represented' by the network, this would not lead us to a representation of the *underlying reality*, i.e. a model from which the function derives.

Now is model-opacity, as we have called it, fundamental or not? We cannot draw a final verdict on this issue, but we may speculate that it is. Above, we always linked the possibility to overcome the other kinds of opacity to their algorithmic transparency, for instance, when benchmarking is used to check whether the purposefully written algorithm performs as desired, or when information theoretic methods are being employed to track a network's evolution. The apparent lack of algorithmic transparency in model-opacity, however, might leave even a Laplacean demon lost in as yet uncharted domains of data. What methods such as the planing performed by Chang et al. (2018) establish is to narrow the space of possible hypotheses down, at least within a conceptual scheme such as quantum field theory, or maybe present day particle physics more specifically. But that still leaves a lot of room, and it is unclear whether all possible hypotheses can be explored in a systematical – let alone algorithmic – fashion. Hence it is equally unclear whether "there are ways to circumvent *any* form of epistemic opacity", as suggested by Durán (2018, p. 107, emph. added).

4 Discussion: A novel phenomenon on any level?

Let us now consider whether any of the opacities discussed here is unique to CS or ML. For the complexity based one, it seems that the identification of social or technological aspects and their say in the opacity of complex code or its implementation renders this sort of opacity a common scientific phenomenon, not a particular feature of CSs: In any complex research task, one has to rely on the standards and expertise of others: specialists will perform and optimize pre-defined sub-tasks, such as calibration of equipment, and unless explicitly requested, one will not get insight into the execution of these tasks, regardless of an involvement of CSs . One usually also does not have deeper knowledge of the equipment one uses:

Measurements performed by a producer can convey some amount of insight, but it also requires some effort to understand the details of a manufacturer's specifications or even test their adequacy.

At this point, we feel the need to emancipate ourselves from, and contrast our findings with those of, Kaminski and Schneider (this volume). Taking crash test simulations as an example, they claim that "opacity occurring here is not due to novelty of the subject or method", but rather, that the following factors "*agglomerate* to epistemic opacity" (emph. added):

1. the social and technical organization of CSs
2. the mathematical background of relevant equations
3. the applied discretization method
4. the application of digital computers, and
5. the quick shift from CSs as objects of research (as in validation and benchmarking) to being a 'valid' research tool.

We will turn to the question of novelty below. For now, we'd like to point out, again, that we do not deny that the combination of these factors creates opacity. However, the above considerations on epistemic relevance, conscious modelling and implementation, as well as testing on several scales and in several complementary ways somewhat lessen the impact of 1., 4., and 5. in HEP. Let us hence focus on 2. and 3.

The main reason for 2., in the case of crash test simulations, is the sensitivity of the 'buckling' of a (thin-wall) structure under a certain load to precise experimental conditions is very high: determining under what conditions the forces that represent a load put onto even a simple structure like a rectangular tube, modeled as a collection of two-dimensional sheets that can bend in three dimensions (so called 'shell elements'), leads to a second order DEQ with sinusoidal solutions. However, the equation's Eigenvalues, corresponding to specific periodic deformations of the tube under a certain load, are determined by the exact material properties and geometry.

Now Kaminski and Schneider find that "This mathematical feature will produce a mathematical opacity when the equations are solved externally." We take this to mean that, because of this high sensitivity to specific conditions, it becomes impossible to track how a specific deformation was reached when the equations are solved as part of a CS.

Many exact proofs of complex mathematical theorems will, of course, be executed by extraordinary mathematicians, and proofs may sometimes remain intransparent, possibly even essentially, for other individuals in the scientific community. As Kripke (1972, p. 159; emph. added) famously put it: "one can learn a mathematical truth a posteriori by consulting a computing machine, *or even by asking a mathematician*."

A nice example is the so called *Feynman-Dyson split* (Kaiser, 2005, pp. 175 ff.): Feynman had introduced his famous diagrams in an intuitive, heuristic fashion, without providing any rigorous mathematical reasons for their functioning on the basis of quantum field theory. This defect was compensated by Freeman Dyson, but in private communications with his Friend Ted Welton, Feynman freely admits to have never thoroughly understood those proofs (cf. ibid., p. 178).

This is well-acknowledged by Kaminski and Schneider, but they also point out that, for the relevant set of equations (and similar ones used in CSs more generally), "there are no experts who understand the dynamics of these systems of equations internally." Further, depending on the edge sizes for the shell elements, i.e., the grain of discretization, the simple tube system shows major differences between steps from one eigenvalue to another, and also differences in steps between the stiffer, narrower side of the tube and the wider, more flexible one. Finally, if the load is incrementally increased above the stable buckling load by means of a Newton method, the system shows strongly non-linear behavior at some point. However, the point at which this behavior occurs, as well as “the complete post buckling behavior”, are influenced by a parameter of the Newton method. Hence, Kaminski and Schneider conclude that “the *technical* opacities of the [discretization method] interfere with the *mathematical* opacity of the instable buckling problem.”

This example – which, in our brief recap, even neglects the additional parameters introduced by the use of digital computers – is paradigmatic for a range of complex problems: An initially simple set of equations leads to rather unpredictable behavior, and this is in part conditioned on the method of discretization and purely computational factors. Even more so, complex mathematical problems will quite often also urge the move even to educated *guesses*: An example already mentioned in Sect. 2.1 is the transfer of factorization schemes for different contributions in proton/proton scattering that is only proven for a small range of scatterings (cf. also Boge and Zeitnitz, 2020, and references therein).

However, addressing point 3. from above, we would like to question whether this really leads to a *novel* kind of *mathematical* opacity that “results from the structure of the systems of equations and their numerical-technical solution”, as claimed by Kaminski and Schneider.

The crucial point is that we do not believe it is right to think of a CS as, strictly speaking, solving the original equations. It is clear that the original equations will be based on our best understanding of the system, and that they will in some sense serve as the basis for various subsequent modeling steps, including discretization methods. But thinking of a CS as literally solving these equations in an approximate fashion means siding with a particular view of CSs (Beisbart, 2012, 2018; Beisbart and Norton, 2012), according to which they can be viewed as arguments, i.e., as extracting the (approximate) consequences of a particular mathematical model.

This view has origins in Humphreys’ (1990) seminal working definition, according to which “[a] computer simulation is any computer-implemented method for exploring the properties of mathematical models where analytic methods are unavailable.” However, Humphreys’ definition was criticized already by Hartmann (1996), who emphasizes the dynamical character of simulations, as well as the fact that CSs are often used even when mathematical solutions are available.

An alternative view has been proposed by one of us (Boge 2019, 2020), which emphasizes the role of various mapping relations from some initial mathematical model to a specific simulation target, as well as the *experimental* character of the use of CSs that is induced by role of the physical computer.

This view, with its weak connection between the original mathematics and the explicit acknowledgement of proper experimental features in simulation, resonates better with views advocated by Winsberg (2010), who emphasizes that CSs have a 'life of their own'; with those of Durán (2020), who urges that, for a proper understanding of how CSs can promote knowledge, it is necessary to take into account the complex internal and external relations of what he calls the 'simulation model'; and with those of Hasse and Lenhard (2017), who underscore that free parameters, as typical of implemented simulation models, are not just boon but also bane of CSs, i.e., allow their flexible use for several contexts, when suitably calibrated to these contexts.

That said, we should *disentangle* the opacity of CSs from mathematical opacity proper. CS have an opacity in their own right and reducing the opacity of some written code will not even necessarily *address* the opacity of mathematical equations that once served as its basis. For example, a perturbation series in quantum chromodynamics is currently not calculable to arbitrary order. So implementing a finite number of Monte Carlo steps that approximately sample the distribution given by the matrix element when calculated to some fixed order does *not* mean *solving* the equations that determine the probability for finding a certain final state from a quantum field theoretical matrix element. Whether the neglected higher order terms distort the picture severely remains, strictly speaking, unknown (though extremely implausible, due to asymptotic freedom), even if we understand perfectly well how the implemented algorithm works, is complied, and physically realized by the computer.

In a way, then, we can turn the arguments by Kaminski and Schneider around: The opacity of the code used for simulation does interfere with the mathematical opacity of the original equations, but 'destructively', not 'constructively'. By using computer code that is consciously written to preserve – a *selected set* of – physical intuitions contained in a certain mathematical model, one replaces the opacity of the mathematical equations (which, without a proper analytical or at least solvable-by-hand numerical solution, persists) by the opacity of a *different* (though connected) model. That latter opacity can be lifted in the indicated ways, i.e., by empirically testing whether the desired, and preservable, properties have been preserved through the various re-modeling steps involved in the numerical design and code-writing. In particular, given that, in crash test simulations, one *knows* that the parameters of the mesh resolution ,of the Newton method, of time resolution, etc., have an impact on the results, it appears that one actually has *insight* into the code: One knows what parameters to pay attention to in order to obtain an empirically adequate model through (iterated) calibration and testing. Hence, one knows that these parameters influence the result, and to what extent they do influence them, or how they must be chosen so as to minimize the impact (as far as possible given economic constraints), becomes an empirical matter.

In other words: divorcing implemented models from original mathematical equations and considering them a method in their own right (not per se a technique for solving equations), the fact that simulation studies can be done to investigate the impact of certain parameters should, in our view, serve as evidence for the (partial) *transparency* of these models, not their opacity.

Hence, we stand by our claim that, *against the background of conscious physics modeling and careful coding*, several levels of testing that establish whether the code is apt for the intended purposes, do reduce the opacity of (HEP-)CSs.

The same reasoning does not apply seamlessly to ML techniques: The results here are adaptations of a given machine to a given task, or, in unsupervised learning, to existing structures in the training data. However, the fact that one utilizes, in the quest for XAI, additional methods external to one's understanding of the underlying algorithms can shed some light on connections to opacities that arise in technology-aided but ML-free research.

Consider, for instance, efforts made for understanding learning machines to those of the ATLAS collaboration for understanding the composition of parts of their detector, e.g. the so called 'insertible *B*-Layer', "a new innermost pixel layer,[...] installed during the shutdown period in 2014" (ATLAS Collaboration, 2017, p. 1). The details of the corresponding study are not terribly important here; in short, data taken in 2015, after insertion, were compared to Monte Carlo simulations in order to reconstruct photon conversion and hadronic vertices in the insertible *B*-Layer, i.e. points of decay of hadrons and photons in virtue of their interaction with the material. Because photon conversion can be reconstructed quite exactly and hadronic decays can be resolved over small angles, these could jointly be utilized to obtain "a detailed radiography of the material, including minute components, e.g. the capacitors mounted on the surfaces of the pixel modules, allowing their location to be determined precisely." (ATLAS Collaboration, 2017, p. 7) This knowledge resulted "in a reduction in the uncertainties associated with the charged-particle reconstruction efficiency determined from simulation." (ibid., p. 1) Hence, before this testing procedure, certain details about the exact material composition of the *B*-Layer were *unknown*. Since these are relevant for the process of reconstructing the tracks of charged particles, it was, in a sense, *opaque* up to that point to what extent the signals received from the *B*-layer were influenced and modified by material imperfections. There is, it seems, no principled difference to the impact of parameters on the outcomes of crash test simulations.

The crucial point is this: while learning machines may be interesting objects of study themselves, in HEP they merely serve as a particular kind of instrument. Hence, just as an increased understanding of the insertible *B*-Layer of the ATLAS detector leads to an increased ability to estimate its use in reconstructing trajectories, will an increased understanding of the processes that lead from inputs to outputs in, say, a neural net increase high energy physicists' ability to estimate its uncertainties, scope of applicability, robustness, or more generally speaking its suitability as a scientific instrument. Opacity induced by the method should, in other words, be seen as continuous with other *technology-induced* opacities in scientific research: while the details obviously differ, it is not a special feature of ML-based analyses that the precise functioning of one's instruments will be opaque in many respects.

What, finally, about model-opacity? Is it, at last, unique to ML? The sobering answer is that even model-opacity is, presumably, not.[86] For compare the finding of a certain signature by some machine to the discovery of the Balmer series in atomic physics: Balmer (1885) discovered a certain pattern in spectral lines that he described systematically by a certain parametrization, but he did not provide any explanation of it. It was not until the advent of the early quantum theory that at least some sort of explanation became available (e.g. Mehra and Rechenberg, 2000, for the historical details), and that required the *ingenuity* of extraordinary physicists such as Niels Bohr, and later Heisenberg and Schrödinger, not merely their computational capacities. Who knows what mathematical and conceptual interventions future physics may urge of us, and whether the right kind of genius is on the horizon?

[86] Note, however, that in the meantime, between the first write-up of this paper and its publication, one of us (FJB) has developed a more fine-grained view that may allow for a specialty in ML after all. See Boge (msa).

Thus, model-opacity is a fundamental problem of scientific inquiry that occurs in exploratory phases, usually when an accepted paradigm fails to provide appropriate guidance w.r.t. a significant range of newly discovered phenomena. Moreover, its presumed non-contingency is basically an invocation of the *problem of unconceived alternatives*:

> The unparalleled breadth, predictive power, precision or other substantive epistemic virtues of some present theories go no distance whatsoever toward showing that we have somehow acquired the ability to exhaust the space of scientifically serious alternatives: indeed, it is in part because these very theories, with the unique advantages over their predecessors they really do enjoy, were themselves previously part of the space of unconceived alternatives that we seem to have every reason to believe that serious, well-confirmed alternatives to our own theories remain presently unconceived.[87] (Stanford, 2006, pp. 44–5)

We acknowledge that this implies a partial shift in focus, as pointed out to us by Andreas Kaminski (priv. comm.). For Humphreys' original account, while formulated w.r.t. a generic process, clearly concerns some scientific *methodology*, such as the process of running a CS (see also Kaminski and Schneider, this volume).

We would like to emphasize, however, that model-opacity is induced, or included, in a special way in ML: Since the above case studies show that ML methods, or particularly DNNs, appear to exploit information implicit in the data (like the complex, higher-order physics variables), which is not necessarily available to the human researchers, they may allow the prediction of new phenomena while leaving any explanatory path to those predictions entirely in the dark (Boge, msa, for a far more detailed treatment). The point remains, however, that *none* of these opacities is entirely *unique* to CSs or ML: All modern research practices are opaque for contingent reasons such as complex mathematics, involved social structure, involved technology, and so forth. Model-opacity, on the other hand, is closely related to, if not an invocation of, the problem of unconceived alternatives, a long-standing problem in the

[87] Indeed, Stanford (2006, p. 45) holds it "possible to imagine cognitive supercreatures who are adept at conceiving of all possible theoretical explanations for a given set of phenomena (or at least all those that they and/or their successors might regard as scientifically serious)[...]." This would render the problem of unconceived alternatives, as well as model-opacity, contingent. So long as we see no evidence of a systematic method such supercreatures could employ, it remains equally possible, however, that no such creatures could exist and model-opacity is indeed fundamental.

philosophy of science—albeit tied here to the specifics of a certain predictive methodology (ML).

5 Conclusions

In this paper, we have shown that ML is associated with an opacity different in kind from that accompanying CSs, and we have coined them opacity induced by complexity (CSs) and opacity induced by the method (ML) respectively. We have then argued that both kinds are contingent on the epistemic conditions of cognitive agents, and that opacity induced by complexity can be overcome by suitable documentation and testing, as done by the ATLAS collaboration at CERN, and that opacity induced by the method may be overcome in a number of ways, e.g. by tracking the evolution of information in stochastic gradient descent.

We have also distinguished these opacities from a *fundamental* one which would pertain to any agent, and suggested that ML (or DL more specifically) is accompanied by another sort of opacity that may be of this kind: model-opacity. This is so because methods such as data-planing can only limit the range of suitable models within a certain theoretical paradigm, but will not positively suggest a given model.

Finally, we have argued that none of the identified opacities are unique to CSs or ML methods: the sources of opacity present in the complexity-induced opacity of CSs all occur in a socially or technologically induced form also in non-CS aided science; and the method-induced opacity associated with ML is really a special kind of technology-induced opacity. Model-opacity, on the other hand, was identified as being basically an invocation of the problem of unconceived alternatives, and thus intimately connected with a long-standing epistemological problem in all of science.

There is, however, a way in which the involvement of ML techniques renders the occurrence of model- opacity special after all. For unlike with human scientist, we cannot converse with a machine or ask for the physical intuitions that guided its discovery. If ML techniques continue on their successful path, this may change the face of science in such a way that discoveries become abundant but are not driven by theoretical results and hypothesis testing anymore. In this way, the scientific process itself may become more opaque in the future.

Acknowledgments The research for this paper was funded by the German Research Foundation (DFG) and conducted as part of the research unit *The Epistemology of the Large Hadron Collider* (grant FOR 2063). We have also profited from comments by audiences at the *SAS Workshop 2018 – Epistemic opacity in computer simulation and machine learning* in Stuttgart, Germany, as well as by members of the of the *PHILETAS* group at ITAS (KIT, Karlsruhe) and the Research seminar at the Institute for Philosophy, in particular Gregor Betz, Thomas von Clarmann, and Michael Poznic. We owe special thanks to Paul Humphreys and Andreas Kaminski for extended commentary and discussion on the manuscript.

References

Aad, G. et al. (2012). Observation of a new particle in the search for the Standard Model Higgs boson with the ATLAS detector at the LHC. *Physics Letters B*, 716(1):1–29.

Andersson, B., Gustafson, G., and Söderberg, B. (1983). A general model for jet fragmentation. *Zeitschrift für Physik C Particles and Fields*, 20(4):317–329.

Andreassen, A., Feige, I., Frye, C., and Schwartz, M. D. (2018). Junipr: a framework for unsupervised machine learning in particle physics. *arXiv*, 1804.09720 [hep-ph].

ATLAS (2010). The ATLAS simulation infrastructure. *The European Physical Journal C*, 70(3):823– 874.

ATLAS Collaboration (2017). Study of the material of the atlas inner detector for run 2 of the lhc. *Journal of Instrumentation*, 12(12):P12009.

ATLAS Computing Group (2005). Computing technical design report. Technical report, CERN. Online: https://cds.cern.ch/record/837738/files/lhcc-2005-022.pdf (checked 11/18).

Baldi, P., Sadowski, P., and Whiteson, D. (2014). Searching for exotic particles in high-energy physics with deep learning. *Nature communications*, 5:4308.

Balmer, J. J. (1885). Notiz über die Spectrallinien des Wasserstoffs. *Annalen der Physik*, 261(5):80–87.
Beisbart, C. (2012). How can computer simulations produce new knowledge? *European Journal for*

Philosophy of Science, 2(3):395–434.

Beisbart, C. (2018). Are computer simulations experiments? and if not, how are they related to each other? *European Journal for Philosophy of Science*, 8(2):171–204.

Beisbart, C. and Norton, J. D. (2012). Why monte carlo simulations are inferences and not experiments.

International studies in the philosophy of science, 26(4):403–422.

Boge, F. J. (2019). Why computer simulations are not inferences, and in what sense they are experiments.

European Journal for Philosophy of Science, 9(1):13. https://doi.org/10.1007/s13194-018-0239-z.

Boge, F. J. (2020). How to infer explanations from computer simulations. *Studies in History and Philosophy of Science Part A*, 82:25–33.

Boge, F. J. (ms). Why trust a simulation? models, parameters and the robustness of experimental results in high energy physics. *unpublished manuscript*.

Boge, F. J. (msa). Two dimensions of opacity and the machine learning predicament. *unpublished manuscript*.

Boge, F. J. and Zeitnitz, C. (2020). Polycratic hierarchies and networks: What simulation-modeling at the LHC can teach us about the epistemology of simulation. *Synthese*. https://doi.org/10.1007/ s11229-020-02667-3.

Brendel, E. (2013). *Wissen*. Berlin: De Gruyter.

Burrell, J. (2016). How the machine ‘thinks’: Understanding opacity in machine learning algorithms.

Big Data & Society, 3(1):1–12.

Chapter 5: Computer simulations, machine learning and the Laplacean demon: Opacity in the case of high energy physics

Carrazza, S. (2018). Machine learning challenges in theoretical hep. *arXiv:1711.10840v2 [hep-ph]*.

Chang, S., Cohen, T., and Ostdiek, B. (2018). What is the machine learning? *Physical Review D*, 97(5):6 pp.

Chatrchyan, S. et al. (2012). Observation of a new boson at a mass of 125 GeV with the CMS experiment at the LHC. *Physics Letters B*, 716(1):30–61.

Clements, A. (2006). *Principles of Computer Hardware*. Oxford, New York: Oxford University Press, fourth edition.

Costanzo, D., Pacheco, A., and Vivarelli, I. (2010). Physics and software validation for atlas. *Journal of Physics: Conference Series*, 219:032064.

de Laplace, P. S. (1902 [1814]). *A Philosophical Essay on Probabilities*. London: Chapman & Hall, Ltd.

Translated from the 6th french edition by Frederick Wilson Truscott and Frederick Lincoln Emory. de Regt, H. (2017). *Understanding Scientific Understanding*. Oxford University Press.

Durán, J. M. (2018). *Computer Simulations in Science and Engineering: Concepts—Practices—Perspectives*. Cham: Springer Nature.

Durán, J. M. (2020). What is a simulation model? *Minds and Machines*, pages 1–23.

Dürr, D., Goldstein, S., and Zanghì, N. (2012). *Quantum Physics Without Quantum Philosophy*. Berlin, Heidelberg: Springer.

Feynman, R. P., Leighton, R. B., and Sands, M. (1965). *The Feynman Lectures on Physics*. Reading, MA: Addison-Wesley.

Fodor, J. A. and Pylyshyn, Z. W. (1988). Connectionism and cognitive architecture: A critical analysis. *Cognition*, 28(1):3–71.

Frigg, R., Bradley, S., Du, H., and Smith, L. A. (2014). Laplace's demon and the adventures of his apprentices. *Philosophy of Science*, 81(1):31–59.

GEANT Collaboration (2016). Physics reference manual. *GEANT 4*, Release 10.4. Gillies, D. (2000). *Philosophical Theories of Probability*. London, New York: Routledge.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. Cambridge (MA), London: The MIT Press.

Griffiths, D. (2008). *Introduction to Elementary Particles*. Wiley, second edition.

Guest, D., Cranmer, K., and Whiteson, D. (2018). Deep learning and its application to lhc physics.

Annual Review of Nuclear and Particle Science, 68:161–181.

Habib, S., Roser, R., LeCompte, T., Marshall, Z., Borgland, A., Viren, B., Nugent, P., Asai, M., Bauerdick, L., Finkel, H., et al. (2015). High energy physics forum for computational excellence: Working group reports (i. applications software ii. software libraries and tools iii. systems). *arXiv preprint arXiv:1510.08545*.

Hartmann, S. (1996). The world as a process. In Hegelsmann, R., Mueller, U., and Troitzsch, K. G., editors, *Modelling and simulation in the social sciences from the philosophy of science point of view*, pages 77–100. Springer.

Hasse, H. and Lenhard, J. (2017). Boon and bane: On the role of adjustable parameters in simulation models. In *Mathematics as a Tool*, pages 93–115. Springer.

Hughes, G. E. and Cresswell, M. J. (1996). *A New Introduction to Modal Logic*. London, New York: Routledge.

Humphreys, P. (1990). Computer simulations. In *PSA: proceedings of the biennial meeting of the Philosophy of Science Association*, volume 2, pages 497–506. Philosophy of Science Association.

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.

Humphreys, P. (2011). Computational science and its effects. In Carrier, M. and Nordmann, A., editors,

Science in the Context of Application, pages 131–142. Dordrecht, Heidelberg: Springer.

Humphreys, P. (2013). Data analysis: Models or techniques? *Foundations of Science*, 18:579–581. James, F. (2006). *Statistical Methods in Experimental Physics*. World Scientific, 2nd edition.

Kaiser, D. (2005). *Drawing Theories Apart*. The University of Chicago Press, London.

Kaminski, A. (2014). Lernende maschinen: naturalisiert, transklassisch, nichttrivial? ein analysemodell ihrer informellen wirkungsweise. In Kaminski, A. and Gelhard, A., editors, *Zur Philosophie der informellen Technisierung.*, pages 58–81. Darmstadt: Wissenschaftliche Buchgesellschaft.

Kaminski, A. (2018). Der Erfolg der Modellierung und das Ende der Modelle. In *Technik–Macht–Raum*, pages 317–333. Springer.

Kaminski, A., Resch, M., and Küster, U. (2018). Mathematische Opazität. Über Rechtfertigung und Reproduzierbarkeit in der Computersimulation. In Friedrich, A., Gehring, P., Hubig, C., Kaminski, A., and Nordmann, A., editors, *Jahrbuch Technikphilosophie 2018: Arbeit und Spiel*, pages 253–278. Nomos Verlagsgesellschaft mbH & Co. KG.

Karaca, K. (2017). The interplay among experiment, simulation and theory at the large hadron collider: A network account of models in high energy physics experiments. Unpublished manuscript, presented at *EPSA17 Exeter*.

Kripke, S. A. (1972). *Naming and Necessity*. Harvard University Press.

Kvanvig, J. (2003). *The Value of Knowledge and the Pursuit of Understanding*. Cambridge: Cambridge University Press.

Larkoski, A. J., Moult, I., and Nachman, B. (2017). Jet substructure at the large hadron collider: A review of recent advances in theory and machine learning. *arXiv:1709.04464v1 [hep-ph]*.

Lenhard, J. (2006). Surprised by a nanowire: Simulation, control, and understanding. *Philosophy of Science*, 73(5):605–616.

Lenhard, J. (2011). Epistemologie der Iteration: Gedankenexperimente und Simulationsexperimente.

Deutsche Zeitschrift für Philosophie, 59(1):131–145.

Lenhard, J. (2019). *Calculated Surprises*. Oxford University Press.

Llorente, J. and Cantero, J. (2014). Determination of the *b*-quark mass m_b from the angular screening effects in the ATLAS *b*-jet shape data. *Nuclear Physics B*, 889:401–418.

Mangano, M. L. and Stelzer, T. J. (2005). Tools for the simulation of hard hadronic collisions. *Annu.*

Rev. Nucl. Part. Sci., 55:555–588.

Marcus, G. (2018). Deep learning: A critical appraisal. *Computing Research RepositoryR*, abs/1801.00631.

Massimi, M. and Bhimji, W. (2015). Computer simulations and experiments: The case of the higgs boson. *Studies in History and Philosophy of Science Part B: Studies in History and Philosophy of Modern Physics*, 51:71–81.

Mehra, J. and Rechenberg, H. (2000). *The Historical Development of Quantum Theory, Vol. 1*. Springer New York.

Morrison, M. (2015). *Reconstructing Reality: Models, Mathematics, and Simulations*. Oxford University Press.

Parker, W. (2018). Climate science. In Zalta, E. N., editor, *The Stanford Encyclopedia of Philosophy*. The Metaphysics Research Lab, Center for the Study of Language and Information (CSLI), Stanford University. online: https://plato.stanford.edu/entries/climate-science/ (checked 12/18).

Pritchard, D. (2014). Knowledge and understanding. In Fairweather, A., editor, *Virtue Epistemology Naturalized: Bridges Between Virtue Epistemology and Philosophy of Science*, pages 315–328. Cham: Springer.

Radder, H. (2009). The philosophy of scientific experimentation: a review. *Automated experimentation*, 1(1):2.

Riggs, W. D. (2003). Understanding 'virtue' and the virtue of understanding. In DePaul, M. and Zagzebski, L., editors, *Intellectual Virtue: Perspectives from Ethics and Epistemology*, pages 203–226. Oxford: Clarendon Press.

Ronen, R. (2017). Why & when deep learning works: Looking inside deep learnings. *CoRR*, abs/1705.03921.

Rudin, C. and Wagstaff, K. L. (2014). Machine learning for science and society. *Machine Learning*, 95(1):1–9.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *nature*, 323(6088):533.

Russell, S., Moskowitz, I. S., and Raglin, A. (2017). Human information interaction, artificial intelligence, and errors. In Lawless, W., Mittu, R., Sofge, D., and Russell, S., editors, *Autonomy and Artificial Intelligence: A Threat or Savior?*, pages 71–101. Cham: Springer International.

Saam, N. J. (2017). Understanding social science simulations: Distinguishing two categories of simulations. In Resch, M. M., Kaminski, A., and Gehring, P., editors, *The Science and Art of Simulation I: Exploring – Understanding – Knowing*, pages 67–84. Cham: Springer International Publishing.

Schembera, B. (2017). Myths of simulation. In Resch, M. M., Kaminski, A., and Gehring, P., editors, *The Science and Art of Simulation I: Exploring – Understanding – Knowing*, pages 51–63. Cham: Springer International Publishing.

Schurz, G. (2008). Third-person internalism: A critical examination of externalism and a foundation-oriented alternative. *Acta Analytica*, 23(1):9–28.

Schurz, G. (2014). *Philosophy of Science. A Unified Approach*. New York, London: Routledge. Schwartz-Ziv, R. and Tishby, N. (2017). Opening the black box of deep neural networks via information.

arXiv preprint arXiv:1703.00810.

Stanford, K. P. (2006). *Exceeding Our Grasp. Science, History, and the Problem of Unconceived Alternatives*. Oxford: Oxford University Press.

Suárez, M. (2003). Scientific representation: Against similarity and isomorphism. *International studies in the philosophy of science*, 17(3):225–244.

Suárez, M. and Cartwright, N. (2008). Theories: Tools versus models. *Studies in History and Philosophy of Science Part B: Studies in History and Philosophy of Modern Physics*, 39(1):62–81.

Sullivan, E. (2019). Understanding from Machine Learning Models. *The British Journal for the Philosophy of Science*. https://doi.org/10.1093/bjps/axz035.

van Fraassen, B. C. (2008). *Scientific Representation: Paradoxes of Perspective*. Oxford, New York: Clarendon Press.

Voss, H. (2013). Classification. In Behnke, O., Kröninger, K., Schott, G., and Schörner-Sadenius, T., editors, *Data Analysis in High Energy Physics: A Practical Guide to Statistical Methods*, pages 153–186. Wiley.

Winsberg, E. (2010). *Science in the Age of Computer Simulation*. University of Chicago Press.

Chapter 6: Chess, Artificial Intelligence, and Epistemic Opacity

Abstract

In 2017 AlphaZero, a neural network-based chess engine shook the chess world by convincingly beating Stockfish, the highest rated chess engine. In this paper, I describe the technical differences between the two chess engines and based on that discuss the impact of the modeling choices on the respective epistemic opacities. I argue that the success of AlphaZero's approach with neural networks and reinforcement learning is counterbalanced by an increase in epistemic opacity of the resulting model.

1 Introduction

Games have always been a welcome area for AI developers to test and develop their newest techniques. Chess is the most famous game in this context and the one that has been studied the most by computer scientists. The fascination for a machine playing chess dates back to the late 18th century when a chess automaton was exhibited throughout Europe. Many of the great minds of early computer science such as Charles Babbage, Alan Turing and John von Neumann devised their own approaches towards a program that would be able to play chess. This culminated in the victory of IBM's "Deep Blue" computer versus the (at the time) reigning world champion, Garry Kasparov, in 1997, which was a major event for the artificial intelligence community. It also started the domination of computer engines in the game of chess; which has today developed so far that a match between a computer and a human player would not be interesting anymore[88] and all top players as well as many amateur players are relying very heavily on computers in their preparation and training (Chessentials 2019). Today's chess engines are very sophisticated programs that include specifically designed search algorithms and evaluation functions, incorporate opening books and endgame databases.

In 2017, 20 years after the success of Deep Blue, a new kind of chess engine has been introduced. AlphaZero is a chess engine based on a neural network created by British AI

[88] Hikaru Nakamura, at the time rated the sixth best player in the world, played a match against a strong chess engine with getting additional material or moves in 2016. He drew three games and lost one (Chabris 2016).

company DeepMind. Its only inputs were the rules of chess and within a few hours of training via playing against itself it became the strongest chess engine in the world. In this paper, I describe this new approach towards chess engines, discuss its differences from other approaches and investigate the hypothesis that the success of this approach with neural networks and reinforcement learning is counterbalanced by an increase in epistemic opacity of the resulting model. An increase in epistemic opacity usually leads to a decrease in our ability to understand and control the resulting model as well as limit our options of learning from it.

In the following sections, I first sketch the development of chess engines (Section 2), and then describe and compare AlphaZero and the strongest traditional chess engine (Section 3). In Section 4, I introduce the concept of epistemic opacity, compare the most advanced classical chess engine with AlphaZero with respect to their epistemic opacities, and discuss the different kinds of opacities that are involved and then conclude (Section 5).

2 History of Chess Engines and AI

In 1770, Wolfgang von Kempelen presented to the Habsburg Archduchess Maria Theresa what would be known as The Turk. It was a machine, which consisted of a wooden man sitting at a cabinet with a chessboard on top of it. The machine was able to move the chess pieces on the board seemingly autonomously. This in itself would not have been very impressive, if it were not for the fact that The Turk was a very good chess player defeating most of its challengers, including famous personalities of the time such as Napoleon Bonaparte, Benjamin Franklin and Charles Babbage. Theories about how the machine works developed quickly, sparked by the same disbelief that led the British author Philip Thicknesse to write: “That an automaton can be made to move the Chessmen properly, as a pugnacious player, in consequence of the preceding move of a stranger, who undertakes to play against it, is utterly impossible” (Thicknesse 1784). As it turned out, he was right for the time being since a small-statured player hidden in the wooden cabinet operated The Turk. Even though the Turk was a hoax it played an important role since it fascinated people with the idea of an intelligent machine and raised questions about the possibilities of thinking machines which are still relevant today (Morton 2015, Standage 2003).

Thicknesses statement was proven wrong for the first time in in the 1950s. Already in the late 1940s both Alan Turing and Claude Shannon (Shannon 1950) created algorithms that were able to play chess. Since no computers with the ability to compute the algorithm were available yet, Turing tried the program in 1951 by calculating everything manually. Also in 1951 Dietrich Prinz, a colleague of Alan Turing, implemented a program which was able to solve mate-in-2 problems. Finally, in 1957 Alex Bernstein, an IBM engineer, implemented a program on a computer for the first time, which was able to play a game of chess (Chessentials 2019).

The first chess engines were not very strong and could be beaten easily by strong amateur players. During the next decades, the quality of the chess engines increased continuously due to developments both in hardware and software and this culminated in the most important event in computer chess history: the IBM computer Deep Blue beating the reigning world champion Garry Kasparov in 1997 in a match of six games. After having won matches against predecessors of the program, Kasparov lost this match 3.5-2.5. Once again, the human-vs-machine setup in chess sparked fascination as well as fear of the developments of artificial intelligence. This time legitimately, as a machine had become strong enough to beat the arguably best chess player of all time (Krauthammer 1997). In the years after the Deep Blue match, a few more matches were played between world-class chess players and chess engines with mixed results, but these matches stopped after 2006, when Vladimir Kramnik, Kasparov's successor as world champion, also lost a six game match. Chess engines now mainly play against each other in Chess Computer Championships. The winner of many of the most recent Computer Chess Tournaments and one of the highest rated Chess Engines is an open-source program called Stockfish. The general programming approach of Stockfish is similar to the one of Deep Blue, using "sophisticated search techniques, domain-specific adaptations, and handcrafted evaluation functions that have been refined by human experts over several decades" (Silver et al. 2018).

In 2017, the AI company DeepMind presented a new kind of chess engine: AlphaZero. Its approach is radically different from the former chess engines; the only input given to AlphaZero were the rules of chess. Using reinforcement learning on a specifically tailored neural network, AlphaZero was then trained by playing against itself for nine hours. The resulting program played a match against Stockfish over 1000 games that it won convincingly (winning 155 games and losing 6). The result in itself is already surprising. Even more surprising to the chess community was the style of the new engine, which is radically different from previous ones. AlphaZero appears to play in a risky attacking style but nearly never runs the risk of losing. Besides that, AlphaZero introduced a number of new motifs and strategies in all stages of the game, which have already been adopted by elite chess players. Therefore, there are more than enough reasons for chess players to delve into the depths of AlphaZero, analyse its games and try to understand the reasons for its success. For philosophers of science, this is an opportunity to gain a better understanding of the success of neural networks and machine learning techniques and to discuss the possible limits inherent to this method.

3 Comparing AlphaZero and Stockfish – a new kind of chess engine

In this section, I compare the techniques used for programming AlphaZero and Stockfish respectively and highlight relevant differences. This will lay the foundation to assess their respective epistemic opacities in the next section.

Chess engines are generally based on two core components. They have a way to evaluate positions and they have a search algorithm that determines which moves are available and in which order they should be considered. For both of these components, Stockfish and AlphaZero use significantly different approaches.

3.1 Stockfish

Stockfish represents chess positions by using a vector that has chess-specific features as elements. These are handcrafted and include elements, which represent for example the pieces each player still has left, the safety of the kings or the pawn structure. The programmers chose those features in cooperation with strong chess players. Each of these features has been included because it has proven relevant in human experience of playing chess and successful in test runs with the engine. Each of these features has a specific weight assigned to it and the evaluation of the position results from a sum of all the features multiplied with their weights. This evaluation is then output in pawn-equivalents. +2.00 for example means that the player with the white pieces has an advantage that is evaluated equivalently to having two extra pawns. (Silver et al. 2018)

In order to figure out what the best move in a specific position is, Stockfish spans a tree of possible developments and evaluates the resulting positions using the evaluation function. This evaluation is only applied to "quiet" positions, i.e. positions without unresolved captures or checks. Therefore, once the desired depth of calculation is reached, there is a quiescence search[89] implemented to resolve all possible tactical elements of the position before the evaluation function is applied to the resulting positions.

Since the amount of positions that have to be evaluated grows roughly by factor 40 for each new level of depth, many heuristics and algorithmic strategies are necessary to cut unnecessary searches and evaluations. The main tool used for this by Stockfish is the alpha-beta algorithm. It is a variant of the minimax algorithm that is common for the evaluation of all kinds of two player zero-sum games. Since it is assumed that both players will act optimally, one cannot simply use the highest value of all the evaluations as the result. Instead, one has to go through all the options to check which of the possible moves of one player leaves the worst options for the other player, since the other player will always choose the best of these options. The alpha-beta search is an optimization of this algorithm that allows eliminating the need to search through all of the different paths. Assume that we already have found a move that guarantees player 1 an equal position, i.e. there is no move from player 2 in response to this move, which leads to any advantage for player 2. From now on, when we consider alternative moves for player 1 and any of the evaluations show an advantage for player 2, we can skip the rest of the search for this particular move, since it is inferior to the move that we already found before. Clearly, this approach works best if we consider the best moves as early as possible. If we would consider the moves in ascending order of strength, we would have to go through all of the possible positions. If instead we always manage to consider the strongest move for each player first, the alpha-beta algorithm would enable us to reduce the nodes that have to be evaluated from x to something close to the square root of x. Therefore, move-ordering is another very important part of the algorithm. Once the move list is generated, it

[89] Quiescence searches are part of traditional chess engines for a long time already. The challenge for the programmer is to extend the search until a position is reached that is suitable to be evaluated by the evaluation function but use minimal resources to do that. For different techniques, see for example Beal 1990.

gets ordered using a number of heuristics, most importantly trying the best move from the previous search first, if the search has been deepened. (Silver et al. 2018, Samuel 1959)

Apart from these very sophisticated algorithms, Stockfish uses an opening book to choose moves in the first phase of the game as well an endgame tablebase that includes the best moves for all positions with six or less pieces left on the board.

The above described techniques are used by most of today's strong chess engines as well as by earlier versions such as Deep Blue.

3.2 AlphaZero

AlphaZero uses none of the techniques described above.

At the core of AlphaZero is a deep neural network that has been trained via reinforcement learning[90]. There is no domain-specific knowledge or data used as input; the training phase has been done exclusively via self-play. The same approach has been used for the games of Shogi and Go successfully[91].

Chess moves are represented in a two-step process in AlphaZero. The first step is to pick up a piece, i.e. identifying the square from which a piece is moved. The second step is deciding which move this piece should make. The input into the neural network therefore contains the 64 squares and all the move possibilities each piece would have on each of these squares. Each square is treated in the same way; this leads to the coding of many illegal moves in specific situations, which are then masked out by setting their probability to zero, reducing the move possibilities to those that are available according to the rules in the actual position. For each square, there are 56 possible queen moves (these moves also code rook, bishop, king and most pawn moves), one to seven squares in all eight directions in which the queen can move. Additionally there are 8 knight moves and 9 possibilities for pawns to promote in a piece different from a queen, either by moving to the last rank or by capturing a piece on the last rank in one of the two diagonal directions. This adds up to an input of 8x8x73 equalling

[90] I cannot provide a description of reinforcement learning or machine learning in general in this paper. For a good introduction see: Goodfellow et al. 2016.

[91] In 2018 Leela Chess Zero, an open-source program with the same approach has been released. By now it has reached a level comparable to Stockfish.

4672 move possibilities in all positions. Additional inputs are needed to represent information about special rules: the castling rights, the number of repetitions of the present position and the move count without progress with respect to the 50 moves rule (Silver et al. 2018).

The output of the neural network is a vector with two numbers for each move. The first one signifies the percentage of search time, which was attributed to this move and its consequences. The second one represents the estimated win probability that the neural network assigns to this move. This means that the neural network has the function of the evaluation function in a classical chess engine. It is trained to assign a probability to each of the possible moves, which represents the win probability when making this specific move. The metric of evaluation is one of the most obvious differences to classical chess engines. This evaluation via probabilities can be interpreted as integrating two aspects, which are not well-represented in Stockfish's evaluation function: The general complexity of the position and the potential risks that result from choosing a specific move. Essentially, this way of representing pays tribute to the fact that the computer cannot calculate all possibly relevant lines and the resulting uncertainty. This will be larger in complex positions with more moves, which cannot be ruled out quickly[92].

The training of the neural network has been done entirely by self-play over a timespan of nine hours. At the beginning of the training phase, all the parameters in the neural network were initialized with random values, leading to seemingly random moves by AlphaZero. Most of these games ended in draws because of the 50-move rule[93], but some of them were decisive and based on these games, the parameters in the neural network were adjusted. Repeating this process for nine hours and 44 million games, the millions of parameters in the neural network were adjusted repeatedly. Via the tuning of these parameters, AlphaZero can represent patterns and strategies. After each decisive game, the parameters in the network

[92] Two behaviours that can be observed in AlphaZero's play can be directly related to this modelling choice. 1) In positions with a large advantage, AlphaZero might simplify the position even on the cost of some of the advantage. While Stockfish would always search for the option that promises the largest advantage, there is no incentive for AlphaZero to win quickly or to achieve unnecessarily large material advantages. This might sometimes lead to moves that seem counter-intuitive for humans such as unnecessary underpromotions. 2) In positions, which AlphaZero evaluates as negative, it might try to complicate the position, making moves that still have a higher degree of uncertainty instead of a well-analysed move, which is evaluated with a very low success probability (Sadler & Regan 2019).

[93] A game of chess ends in a drawn, if for 50 moves in a row, no pawn is moved and no piece is captured.

are updated to evaluate each of the positions of the game as better or worse depending on the outcome of the game (Sadler & Regan 2019).

We can conclude that there are two major differences in the architectures of AlphaZero and Stockfish. The evaluation of positions is approached with different metrics. Stockfish evaluates positions in pawn equivalents; AlphaZero assigns win probabilities to positions. The most important difference lies in how the evaluations are reached. Stockfish uses handcrafted domain-specific knowledge in its evaluation function; AlphaZero reaches the evaluation of the position through its neural network, which has been trained by self-play and without the input of any domain-specific knowledge except for the rules of the game.

4 Epistemic opacity

> *“Whilst we cannot understand exactly how AlphaZero is thinking, we can explore the ways in which AlphaZero generates its innovative and active plans, and how it conducts its ferocious attacks through analysing its games.” (Sadler & Regan, p. 74)*

It is certainly not appropriate to associate the calculations that lead AlphaZero to make decision with thinking. If interpreted a bit more metaphorical, this quote gives a first answer to one of the main questions of this paper: What can we learn about how AlphaZero works and how is it different to what we can learn about Stockfish or similar other chess engines?

Since AlphaZero managed to beat Stockfish, it would be very interesting and potentially beneficial for our understanding of the game of chess, to learn on what AlphaZero bases its decisions. It seems to be a reasonable assumption that it must have something encoded about the game of chess that goes beyond the handcrafted domain-specific features, which have been encoded in Stockfish.

The thesis that I want to defend here is that the introduction of a deep neural network and machine learning led to a different kind of epistemic opacity in AlphaZero than the one in Stockfish. This new kind of epistemic opacity prevents us from confidently identifying on what AlphaZero bases its decisions.

4.1 Defining epistemic opacity

At first, it seems counter-intuitive that either Stockfish or AlphaZero can be epistemically opaque in any way. Both are based on algorithms, which are determined processes. For each of the calculations that are done in any of the chess engines, it is always clearly defined which rule or calculation has to be applied next. In principle, all of these calculations could be done by pen and paper or printed out.

Nevertheless, in the philosophical debate epistemic opacity is discussed very prominently. Especially in the context of software that is based on machine learning techniques as in the case of AlphaZero, their potential black box-nature is a widely discussed topic. The introduction of such methods into the scientific practice raises questions about the aims of science and necessity of explanations or understanding.

Let us take the most commonly used definition of epistemic opacity by Paul Humphrey as a starting point to understand what is typically meant by epistemic opacity and why it is useful discussing it in the context of computer models and machine learning.

> *"[A] process is epistemically opaque relative to a cognitive agent X at time t just in case X does not know at t all of the epistemically relevant elements of the process. A process is essentially epistemically opaque to X if and only if it is impossible, given the nature of X, for X to know all of the epistemically relevant elements of the process." (Humphreys 2011, p 139)*

One of the potential sources of opacity is the sheer amount of calculations that are done by Stockfish or AlphaZero during a game of chess, which lead to the evaluations of positions and the decision for a specific move. Stockfish calculates up to 60 million positions per second, so it is quite clear that no human cognitive agent will ever go through all of the calculations that are made during one entire game for example. Even though AlphaZero calculates less positions per second (Silver et al. 2018), the amount of calculations still exceeds everything a human agent would be able to go through in a reasonable time. Regarding AlphaZero, one can additionally argue that the steps during the training phase are epistemically relevant elements of the process, since they determine the final values of the parameters in the neural network and thereby have a significant impact on the process. This adds a large number of calculations to the epistemically relevant elements[94]. It is therefore clear that the basic kind of epistemic opacity that Humphreys defines in the first sentence of his definition is present in both Stockfish and AlphaZero.

Are these opacities essential in the sense of Humphreys' definition? To argue for this, one has to show that it would be impossible for the cognitive agent to go through all of these calculations because of their nature. Let us take a human as the cognitive agent X. He is limited through his life span and if we assume a discrete amount of time that is needed for each of the calculations, we can determine a maximum number of calculations that one human could possibly do and thereby determine a threshold for this opacity to become essential. This approach is however not very informative about the nature of the source of opacity. It seems somewhat arbitrary that there should be some number of calculations x, which are necessary for a process, for which the opacity is not essential; if at the same time, adding just one more calculation would make the opacity essential.

As has been argued in Boge & Grünke (2020), there might be another useful differentiation between types of opacities, namely "contingent epistemic opacity" and "fundamental epistemic opacity". Both epistemic opacity and essential epistemic opacity, as defined by Humphreys, are contingent on the nature of the cognitive agent. Fundamental epistemic opacity on the other hand refers to opacities which are grounded in the nature of the process rather than the nature of the agent:

> *"[A process is fundamentally epistemically opaque, if] given any agent with any nature, at no time will the agent be able to obtain all relevant pieces of information about the process." (*Boge & Grünke 2020*, p. 9)*

A thought experiment shows that the opacity, which is caused by the amount of calculations, is contingent. Consider the Laplacean demon, understood as an entity with unlimited cognitive resources, which would be able to perform as many calculations as necessary

[94] There is a debate about what should count as epistemically relevant. Durán (2018) for example points to the fact that a limited amount of information about an algorithm might be enough for the justification of results. In this paper, I follow the more widespread approach to count all parts of the process as epistemically relevant to it. The reduction of epistemically relevant elements requires background information. There is not enough of it available yet in this case, therefore using the wide approach seems to be more appropriate.

without using up any time (de Laplace 1814). For it, the algorithmic transparency of the processes would be enough to render them epistemically transparent (Boge & Grünke 2020).

With respect to this source of opacity, AlphaZero has introduced no different kind of opacity.

4.2 Model-opacity as fundamental opacity

The definition by Humphreys shapes the concept of epistemic opacity as something that is connected to processes. This makes it natural to look for sources of opacities, which may arise in the process of coming to the results via a string of calculations or the training phase of the neural network.

Sadler & Regan (2019) focus on the question "how AlphaZero is thinking" (p. 74), but in order to learn the new things, which AlphaZero might have discovered about chess that make it more successful than traditional chess engines, we need to understand *what* AlphaZero is thinking (thinking must be understood metaphorically again, of course). All of AlphaZero's decisions are based on the neural network that has been trained. By tuning the parameters of the neural network, AlphaZero represents features of the positions, similarly to the way features of positions are represented in the handcrafted evaluation function of Stockfish. In the case of AlphaZero however, it is not clear which features are actually represented in the neural network, since they are developed during the training phase and not preselected through the programming. It is possible that AlphaZero learns features, which have been chosen for Stockfish, such as safety of the king or pawn structure. It could also be the case that AlphaZero learns features, which are a lot more complex and far removed from the human way of describing concepts in chess. This uncertainty about what features are modelled in the neural network may constitute a different kind of opacity. It is not process-based but rather concerns the connection between the neural network and the real-world phenomenon, so it seems useful to call it *model-opacity*[95].

[95] Model-opacity is discussed in Boge & Grünke (2020) for an example from high-energy physics. Sullivan (2019) uses the same term for the "complexity and black box nature of a model" (p.1). She introduces the term "link uncertainty" to describe the situation, that there is not enough empirical or scientific evidence to prove the connection between the model and a real-world phenomenon.

One natural approach of trying to overcome part of this opacity is a classical experimental one[96]. You come up with a hypothesis about the behaviour of the system and collect empirical evidence for or against it. The main source for empirical evidence in the case of AlphaZero are the games that have been published. Sadler & Regan (2019) analyse the games and try to identify features in AlphaZero's play. They show examples of well-known strategies, which have been adopted by AlphaZero, as well as positions in which AlphaZero does not follow any of the classical plans developed by humans and programmed into Stockfish[97]. However, the complexity of chess makes it impossible to isolate any specific feature in almost all cases. Each position can be described by many different features and their interactions. At the same time, these features cannot be deduced directly from the rules of chess but are human descriptions of patterns that have been recognized. The number of ways to describe features of a chess position is infinite and many might be functionally nearly equivalent but differ in some very specific ways. Therefore, in addition to the problem that we cannot isolate specific features, it is impossible to be sure that we have conceived all of the potential hypothesises. Even unlimited resources would not solve this problem. There is no algorithm that could solve this problem and therefore there is no algorithmic transparency, which would render this opacity contingent. Model-opacity therefore seems to be fundamental (cf. Boge & Grünke 2020 for a similar case).

4.3 Discussion of the opacities

The contingent opacity caused by the amount of calculation steps is similar for both Stockfish and AlphaZero. Even though there are differences in the concrete amount of calculations and AlphaZero has the training phase of the neural network, which influences the results and does not have a complementary part in the Stockfish architecture, the nature of these opacities is the same. All of these calculations are clearly defined and determined processes that can be solved if enough resources are available.

[96] Another approach is to find positions in which engines obviously misevaluate and try to derive information about the system from that (similar to adversarial networks in image classification). There are a number of constructions of positions with enormous material advantage for one player in which a human can quickly see that there is no way to achieve any progress but the computer nevertheless evaluates the position as very advantageous.
[97] The minority attack in the Carlsbad structure is used as a prominent example.

As described above, this is not the case for model-opacity. However, model-opacity does not exist for Stockfish. It does not exist because the modelling for Stockfish was intentional and goal-orientated. The modellers can be asked about the reasons and intentions for the implementation of specific features. This does not mean that Stockfish never makes moves which come unexpected to the modellers. No one who modelled parts of Stockfish has a complete comprehension of the interaction of all parts of the program and can calculate all the consequences of these interactions and therefore no one can make a confident prediction. Stockfish outperforms all human players in the game of chess after all. This is parallel to AlphaZero but can be explained with contingent opacity. What is possible when Stockfish makes an unexpected move is to analyse the reasons how the decision to make this specific move came about. By reviewing which evaluations led to the move and which features were evaluated in what way, a reconstruction of the decision process is possible. This is exactly what seems to have happened in the match between Deep Blue and Garry Kasparov. During the first game of the match, a bug in the code led to an unexpected move by Deep Blue. After the game, the programmers of Deep Blue could track down the reason why this move was chosen and fixed the bug (Anderson 2017). If this would happen with AlphaZero, the programmers would not know how to change the code of AlphaZero. There is no way to predict what the consequence of changing a single weight in the neural network would be for example. There is no way of confidently knowing about the way specific features are represented in AlphaZero's neural network and consequently they cannot be manually altered with a specific goal in mind.

This opacity of AlphaZero is of a different kind than the contingent ones and it does not only prevent humans from intentionally altering the program in an advantageous way but also prevents them from isolating and understanding the way in which AlphaZero represents chess, and which features of the game it chose in its neural network.

5 Conclusion

In this paper, I described the development of artificial intelligence in chess with its present-day culmination: the introduction of the neural network-based AlphaZero in 2017. I highlighted its differences from Stockfish, one of the highest rated traditional chess engines.

The two most significant differences are the modelling process and the metric of evaluation, most notably the creation of the neural network of AlphaZero without any domain-specific knowledge about chess except for its rules.

The different modelling processes lead to different epistemic opacities. Both Stockfish and AlphaZero are epistemically opaque in a contingent way due to the very large number of calculations necessary during the training phase of AlphaZero as well as during the play phase for both Stockfish and AlphaZero. AlphaZero additionally also has model-opacity, which seems to be a fundamental kind of opacity. This opacity originates from the method that is used for creating the neural network: machine learning or, more specifically, reinforcement learning. Contrary to the modelling process of Stockfish, the modelling is not done intentionally by a human programmer, but through reinforcement learning – a statistical approach. This seems to make it impossible to connect the "reasoning" of AlphaZero to human reasoning in chess.

This fundamental opacity is not problematic in chess. Human chess players can still follow AlphaZero's games, use it as an inspiration and develop ideas and concepts from it, which they can integrate in their own games. For applications in domains other than games however, it is potentially ethically very problematic if human agents base their decisions on the output of a neural network with fundamental opacity.

Acknowledgements

The research for this paper was partially conducted as part of the research unit The Epistemology of the Large Hadron Collider, funded by the German Research Foundation (DFG; grant FOR 2063). I want to thank the participants of the Budapest Workshop on Philosophy of Technology 2019 for a very helpful discussion on the topics of this paper. I would also like to thank an anonymous reviewer for helpful suggestions on improvements of the original manuscript.

References

Anderson, Mark Robert (2017). *Twenty years on from Deep Blue vs Kasparov: how a chess match started the big data revolution.* [online] Available at: http://theconversation.com/twenty-years-on-from-deep-blue-vs-kasparov-how-a-chess-match-started-the-big-data-revolution-76882

Beal, Don F (1990). "A generalised quiescence search algorithm." *Artificial Intelligence* 43, no. 1: 85-98.

Boge, Florian, and Paul Grünke (2020). "Computer Simulations, Machine Learning and the Laplacean Demon: Opacity in the Case of High Energy Physics", forthcoming in Resch, Kaminski, and Gehring (Eds.), *The Science and Art of Simulation II,* Springer (expected 2020).

Chabris, Christopher (2016). *The Surprising Return of Odds Chess*. [online] Available at: https://www.wsj.com/articles/the-surprising-return-of-odds-chess-1461339115

Chessentials (2019). *History Of Chess Computer Engines*. [online] Available at: https://chessentials.com/history-of-chess-computer-engines/

de Laplace, P. S. (1902 [1814]). *A Philosophical Essay on Probabilities*. London: Chapman & Hall, Ltd.

Translated from the 6th french edition by Frederick Wilson Truscott and Frederick Lincoln Emory.

Durán, J. M. (2018). *Computer Simulations in Science and Engineering: Concepts—Practices—Perspectives*. Cham: Springer Nature.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville (2016). *Deep learning*. MIT press.

Humphreys, P. (2011). Computational science and its effects. In Carrier, M. and Nordmann, A., editors, *Science in the Context of Application*, pages 131–142. Dordrecht, Heidelberg: Springer.

Morton, Ella (2015). *The Mechanical Chess Player That Unsettled the World*. [online] Available at: https://slate.com/human-interest/2015/08/the-turk-a-chess-playing-robot-was-a-hoax-that-started-an-early-conversation-about-ai.html

Sadler, Matthew, and Natasha Regan (2019). "Game Changer." *New in Chess*.

Samuel, Arthur L. (1959). "Some studies in machine learning using the game of checkers." *IBM Journal of research and development* 3, no. 3: 210-229.

Schwartz, Oskar (2019). *Untold History of AI: When Charles Babbage Played Chess With the Original Mechanical Turk.* [online] Available at: https://spectrum.ieee.org/tech-talk/tech-history/dawn-of-electronics/untold-history-of-ai-charles-babbage-and-the-turk

Shannon, Claude E. (1950). "XXII. Programming a computer for playing chess." *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 41, no. 314: 256-275.

Silver, David, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot et al. (2018). "A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play." *Science* 362, no. 6419: 1140-1144.

Standage, Tom (2003). *The Turk: The Life and Times of the Famous Eighteenth-Century Chess-Playing Machine.* New York: Berkley Trade.

Sullivan, Emily (2019). “Machine Learning and Understanding”. forthcoming in *British Journal for the Philosophy of Science.* Available at: http://philsci-archive.pitt.edu/16276/

Thicknesse, Philip (1784). *The Speaking Figure, and the Automaton Chess-player, exposed and detected*. London: Stockdale. 8vo. pp. 20.