Nat Comput (2018) 17:271–281
https://doi.org/10.1007/s11047-017-9643-0

Model-based computation

Cameron Beebe[1,2]

Published online: 4 October 2017

Abstract A brief analysis of analog computation is presented, taking into account both historical and more modern statements. I show that two very different concepts are tangled together in some of the literature—namely continuous valued computation and analogy machines. I argue that a more general concept, that of *model-based* computation, can help us untangle this misconception. A two-dimensional view of computation is offered, in which this model-based dimension is orthogonal to the dimension concerning the type of variables used in components. I argue that this is a useful framework for assessing alternative computing devices and computational claims in an expanding landscape of computation.

Keywords Model · Computation · Analog

1 Introduction

The scope of this present article is not formal, but conceptual.[1] I wish to provide a general discussion on the notion of computation, motivated by the fact that there seems to have been some confusion present in much of the literature concerning analog computation. By an analysis of this confusion and the wider 'computational landscape', I hope to contribute to our understanding of some recent claims by introducing what I call *model-based computation*. Model-based computation can be seen as a distinct 'dimension' with which to evaluate devices in a wider computational landscape, and it allows us to see the flaws in the confused argumentation present in the literature cited. The arguments are largely couched in an orthogonal dimension, the dimension concerning variable types of components, rather than the model-based dimension. Furthermore, I argue that this two-dimensional view is a natural extension for current notions of computation, and is well-motivated from the analysis provided in the first sections of this present work.

A first step in this project is to provide evidence that there is a conceptual confusion present in discussions of analog computation. This helps establish what analog computation is *not*, and motivate the discussion in subsequent sections of what it *is*—and how a more general two-dimensional notion of computation accommodates it. We begin with two statements from Nielsen and Chuang's textbook of quantum information theory, which I quote at length for the unfamiliar reader:

> In the years since Turing, many different teams of researchers have noticed that certain types of analog computers can efficiently solve problems believed to have no efficient solution on a Turing machine. At first glance these analog computers appear to violate the strong form of the Church–Turing thesis. Unfortunately for analog computation, it turns out that

✉ Cameron Beebe
cameron.beebe@campus.lmu.de

[1] Research Center for Neurophilosophy and Ethics of Neurosciences, Ludwig-Maximilians-Universität, Schellingstrasse 10, 80799 Munich, Germany

[2] Munich Center for Mathematical Philosophy, Ludwig-Maximilians-Universität, Munich, Germany

[1] An earlier version of this paper originally appeared in Beebe (2016). This current paper has extended and improved upon the original by incorporating several new references, updating definitions and clarifying the argument structure, and by providing additional discussion of important points throughout.

when realistic assumptions about the presence of noise in analog computers are made, their power disappears in all known instances; they cannot efficiently solve problems which are not efficiently solvable on a Turing machine. This lesson — that the effects of realistic noise must be taken into account in evaluating the efficiency of a computational model — was one of the great early challenges of quantum computation and quantum information, a challenge successfully met by the development of a theory of quantum error-correcting codes and fault-tolerant quantum computation. Thus, unlike analog computation, quantum computation can in principle tolerate a finite amount of noise and still retain its computational advantages. (Nielsen and Chuang 2010, p. 5)

One might suspect that quantum computers are just analog computers, because of the use of continuous parameters in describing qubit states; however, it turns out that the effects of noise on a quantum computer can effectively be digitized. (Nielsen and Chuang 2010, p. 164)

There seems to be an assumed notion of analog computation as a delicate and noise-intolerant business. Even discussions like those from Turing (1950, Sect. 5) concerning the sensitivity of physics to variations in initial conditions might be taken as supporting such a notion. A careful reading reveals that this statement can only be understood in support of the noise tolerance of discrete state machines, and not a statement claiming that any analog computation is noise intolerant. It might be possible to infer something about noise intolerance in some continuous variable computational devices, but that is a separate question (and there might be practical ways of avoiding the problem we do not want to close ourselves off to).

Do these statements about sensitivity to noise actually have relevance for evaluating analog computation generally? I argue that an affirmative answer stems from the core misconception that continuous valued 'organs' (that is, components performing specialized functions) are not only essential to analog computation (they are not) but that a device which has such organs is *synonymous* with analog computation.[2] This is simply not the case. Continuous values are neither necessary nor sufficient to characterize analog computation.

The reader may also wish to see argument 6 from Scott Aaronson's page on skeptics of quantum computation, where he claims "We know that analog computers are not that reliable, and can go haywire because of small errors." Aaronson proceeds to respond to the question of "why a quantum computer should be any different, since you have these amplitudes which are continuously varying quantities." In his response, he makes the very conflation at question here, namely that analog computation is synonymous with continuous value computation.[3]

Additionally, there is not just *one* kind of physical system that can constitute an analog device, meaning that such statements of an analog computer "going haywire" could only be safely interpreted on this level as referring to a particular architecture—but clearly it is meant as a general statement about analog computation. We could take these statements to refer to some formal generalization of the concept of analog computation, but we will see that such conclusions about the weakness (or effectiveness) of an analog computer must be evaluated with respect to a particular *model*.

These claims against analog computation are not supported by the analysis of analog computation offered in Sect. 2, where I introduce what I think is a much clearer conception of analog computation. I then proceed in Sect. 3 to outline some thoughts on what I call 'model-based computation' respecting this conception. Afterwards, I evaluate two discussions in light of this notion of model-based computation. The first includes computational claims about the brain and the current notion of hierarchical generative models in cognitive science. We will see in Sect. 4 that hierarchical generative models seem to describe model-based computation as outlined in this present work. The second discussion in Sect. 5 will focus on analog models in physics, in particular the notion of analog simulation recently put forth in Dardashti et al. (2015).

Relevant aspects of the notion of model-based reasoning will be discussed in Sect. 6, where we see the relationship with ideas already present in cognitive science. This relationship is arguably unsurprising, given that our notion of computation has historically been informed by the inference capabilities of an intelligent agent. A computer was, after all, initially a term used to refer to a particular desk job. In the concluding Sect. 7 I will draw attention to what I think is the importance of this discussion for the developing market place of alternative computing, and summarize the view of a two-dimensional landscape of computation.

2 What is analog computation?

Rather than being defined by the continuity of parameters, an analog computer can be defined through a literal treatment of the word *analogy*—and in fact Neumann (1963, p. 293) among others even refers to two classes of computing

[2] We will see shortly that von Neumann, among others, used the term 'organ' for computational components.

[3] http://www.scottaaronson.com/democritus/lec14.html.

machines, analogy and digital machines. The term *analogy machines* sounds much different to our modern ears than analog computer, but I argue it more accurately represents this area in the landscape of computation. Particularly in modern computer science where alternative or specialized computing devices have become more common, it is arguably worthwhile to have a clearer conceptual overview of this landscape.

I argue that by clearing up the issues mentioned above, we can see two dimensions characterizing the landscape. The first dimension has to do with the types of variables processed by computational components (or organs). The other is the extent to which the structure of a device *models* a target problem. The first dimension is arguably uncontroversial, and I do not focus on justifying its incorporation in the two-dimensional view. The second dimension about models warrants significant discussion, and forms the bulk of the rest of this present work.

Before going into model-based computation generally, it is helpful to first analyze deeper the notion of analog computation. Bernd Ulmann, distilling contributions from many previous authors on the subject, has provided us with a clear assessment of what analog computation *is*. I quote at length since I believe his comments are very informative and are incapable of compression without loss.

> First of all it should be noted that the common misconception that the difference between *digital computers* on one side and *analog computers* on the other is the fact that the former use discrete values for computations while the latter work in the regime of continuous values is wrong! In fact there were and still are analog computers that are based on purely digital elements. In addition to that even electronic analog computers are not working on continuous values — eventually everything like the integration of a current boils down to storing (i.e., counting) quantized electrons in a capacitor.
> If the type of values used in a computation — discrete versus continuous — is not the distinguishing feature, what else could be used to differentiate between *digital* and *analog* computers? It turns out that the difference is to be found in the structure of these two classes of machines: A digital computer in our modern sense of the word has a fixed structure concerning its constituent elements and solves problems by executing a sequence (or sequences) of instructions that implement an algorithm. These instructions are read from some kind of memory, thus a better term for this kind of computing machine would be *stored-program digital computer* since this describes both features of such a machine: Its ability to execute instructions fetched from a memory subsystem and working with numbers that are represented as streams of digits.
> An analog computer on the other hand is based on a completely different paradigm: Its internal structure is not fixed — in fact, a problem is solved on such a machine by changing its structure in a suitable way to generate a *model*, a so-called *analog* of the problem. This analog is then used to *analyze* or *simulate* the problem to be solved. Thus the structure of an analog computer that has been set up to tackle a specific problem represents the problem itself while a stored-program digital computer keeps its structure and only its controlling program changes. (Ulmann 2013, p. 2)

Going back to von Neumann, we find the beginning of the next most essential aspect of analog computation (and computation in general)—that computation depends on the *use* of a system. Even though the components in a device might be ultimately continuous, it depends on how we intend to use the system. We also see evidence that the confusion concerning analog computers has been around for quite some time:

> The electromechanical relay, or the vacuum tube, when properly used, are undoubtedly all-or-none organs. Indeed, they are the prototypes of such organs. Yet both of them are in reality complicated analogy mechanisms, which upon appropriately adjusted stimulation respond continuously, linearly or non-linearly, and exhibit the phenomena of "breakdown" or "all-or-none" response only under very particular conditions of operation. (Neumann 1963, pp. 297–298)

We should be careful in parsing this particular quote, since von Neumann uses 'analogy' and 'continuously' in the same sentence. It seems that here he has also conflated analogy with continuity, although it is unclear whether he means something more than continuous. In other places he seems to maintain a clearer distinction, but in any case the issue has not gotten clearer in the more modern statements quoted earlier. However, what we see is that 'proper use' is essential to defining computation. The close relationship historically between analog computation and modeling in science is also discussed by Care (2010) in depth. These ideas will be discussed throughout the paper, but for now we can state more accurately what we mean by an analog computer.

Definition 1 (*Analog computer*) An analog computer is a device whose internal structure is malleable and can be set up to have similarities to aspects of the class of problems it is used to solve. Additionally, these similarities by themselves should be sufficient to form a *model* of the relevant class of problems. In our proper use of the device, the organs involved are interpreted by the model to function in a way that is consistent with our understanding of what would be required to solve the target problems.

While some analog computers under this definition can indeed be considered as (ideally) implementing differential equations or having continuous organs relevant to our purposes, this must be recognized as only a subset of potential uses of such a computer. In other words, the definition does not explicitly endorse smoothness or rule out digital systems. What is more important for the notion of analog computation, and for developing a richer conception of computation, is that the user and the architecture both play important roles in their relationship to a *model*.[4] The user has to develop a model, or recognize similarities, or utilize analogical reasoning to set up the system in such a way that it can solve the problems at hand.

Our view of the architecture reflects this modeling procedure, meaning that as von Neumann notes an 'all-or-nothing' organ might be liable to be characterized under other usages as a more or less continuous valued organ. What should be clear at this stage is that analog computation utilizes a *model* to frame the use of the device, not unlike how models are used in other areas of science to represent sets of properties and relations relevant to a given inquiry. For analog computers, this seems to have historically been models incorporating similarity and analogy. However, I argue this is just one type of a more general category of what can be called *model-based computation*. At this point I diverge slightly from Ulmann's statements, although the central premise is, I think, present in his work already quoted and thus I am offering more of a naturally implied extension than a meaningful divergence.

Before moving on, however, we can state that there may be particular objections to construing analog computation in the manner done here. One might say that 'analog' has taken on a new meaning, and that for all intents and purposes analog computation is just defined nowadays as computation with continuous variables. I must insist on disagreeing with this approach to redefining terminology, since it leads to unnecessary confusion. Piccinini and Bahar (2013) throw out the very idea of analog-model based computation that Ulmann has brought to our attention (and which I am agreeing with here):

> In another sense, "analog" refers to representations that bear some positive analogy to what they represent. There is evidence that nervous systems contain and manipulate analog models of their environment [...] But analog *models* need not be represented and manipulated by analog *computers*. (Piccinini and Bahar 2013, p. 466)

The authors are correct if what they mean by analog computer is just a computer with continuous variables. However, I find the arguments and examples advocated for here (and extensively in Ulmann's work) to convincingly establish that analog computation is not (and has not even historically been) identical to computation with continuous variables. With this in mind, the authors might be sympathetic to the solution advocated for here: disentangling continuity and analogy into two separate dimensions of the general notion of computation. One dimension is that of the types of variables manipulated in components. The other dimension concerns the extent to which a computational device can *model* a target.

Later, Piccinini (2015, Sect. 12) indeed seems to recognize the confusion in the literature with continuous variables, although his discussion in the chapter largely focuses on electronic analog computers and presumes real variables. He also claims Piccinini (2015, p. 199) that the "notion of an analog model [...] is orthogonal to the notion of analog computation." He seems to acknowledge that this is not historically true, but he thinks it is conceptually accurate. If he means by this statement that the model is orthogonal to the types of variables manipulated, then we are in agreement. If he instead means that the model is orthogonal to what it means to compute, my account is in disagreement since I think it is clear models are representations.[5] Taking into account his view that computation does not require representation seems to support the latter interpretation, and goes against the model-based notion of computation presented here.

3 Model-based computation

The notion of model-based computation is easily inferred from those definitions already provided for analog computation by Ulmann. The notion is just slightly more general, in that the model used may or may not incorporate similarities or analogies to the extent that analog models do.[6] That is, even if there are similarities in the device, these similarities may not be sufficient by themselves to form a model of the target problem class. Some other parameters might, for example, be invoked to make the model work even though it is unknown whether such a

[4] Thus, in the remainder of this article, the reader should note that when I use the term 'analog', even as an adjective, it does not refer to continuity.

[5] Also, Care (2010) advocates a deep relationship between analog computing and modeling in science supported by an in depth historical analysis.

[6] Although it may be an open question whether all models are in fact rooted in analogy or similarity, I do not focus on such an argument here.

parameter corresponds to the target. It could even be that the parameter is *known* to be dissimilar to the target, but the model is sufficient nonetheless for our uses. Analog computation is then a special case more accurately thought of as shorthand for analog-model computation. It might be that good examples of model-based computation are in fact using analog models, but it is arguably a small class within the computational landscape compared to any potential model-based computation—if only for the reason that analog models are a restricted class of models in general.

But what is a model? This question has been widely addressed in the philosophy of science community, and a few brief notes might be helpful before moving forward. There are a variety of different kinds of models which are used in science. There are toy models, idealized models, scale models, mathematical models, and many more kinds of models. Each of these kinds may have overlap with other kinds, they are not exclusive of each other. All models, it seems, need a target object or set of data which is to be represented or accounted for in some way in the model. See Frigg and Hartmann (2012) for more on models in science. Model-based computation may involve many of the same aspects as other models in science.

Definition 2 (*Model-based computer*) A model-based computer is a device which may have a malleable internal structure, and which can represent aspects of the class of problems it is used to solve. The representations should be sufficient to form a model of the target problem class. Under proper use, the organs in the device can be interpreted by the model to function in a manner that we take to solve the target problems. This may or may not be consistent with our understanding of the target problem class.

It is useful to go one step further in this section, to discuss model-based simulation. This is helpful before encountering analog simulation in later sections. Model-based simulation is a refined form of model-based computation, in which the dynamics of the device are relevant for the user or target problem (as opposed to just a functional relation). Generally, the dynamics of a model-based computer may or may not model what we know about the dynamics of the target system. Simulation operates on a richer model that deems relational aspects (such as temporal or dynamical relations) of the device relevant. One can have static features represented in a model which, after use, has an output which functionally represents a useful computation concerning a target problem. However, the dynamics of using the model may be irrelevant to the kind of dynamics present in the target system. In this case we would not say that there is model-based simulation present.

Just reading the output of a slide rule, for example, does not seem to involve simulation but just accomplishes a computation with the model. Take two equal length sticks with logarithmic scales on them lined up side by side. Multiplication can be calculated by sliding one of the sticks relative to the other by a factor. That is, 2 times 4 could slide one stick by 2 on the logarithmic scale (representing a multiplication of 2). Then, one looks up the other factor and reads off the corresponding value on the other stick. In this case, 4 would be lined up with 8, the result of the calculation.[7]

The dynamics of sliding the stick does not seem to model an algorithm for multiplying some integers. The model in this case involves not only the physical ruler, but also the reasoning and mathematics involved to create the scales encoded in the ruler. The preparation of the computing device has utilized pre-computed knowledge [i.e. $\log(xy) = \log(x) + \log(y)$] to functionally output values consistent with an algorithm for multiplication, but the sliding dynamics are not relevant for the computation. I think it is quite reasonable to expect that model-based computation, in general, does not necessarily include relevant dynamics with the target system. When it does, a stronger notion of model-based simulation is applicable. We will see later an example of analog simulation in which the dynamics are relevant *and* similar.

As this is a relatively simple example, we can say a bit more about what a model consists here. For the slide ruler, our target problem is modeled by a set of input operations which obtain, upon a functional relation, a desired output. That is, our model is the set of representations and operations upon such representations that are used. The end step after a specific procedure can be interpreted as our desired result, that which is to be computed. If our problem is to multiply two integers, a slide rule encodes two logarithmic scales. We operate upon such scales by sliding (adding) one factor, and then reading out the number mapped from the factor on the first stick to the number on the other stick.

Consider that a plastic model of a molecule consists of all aspects standing in a representational relation to an *actual* molecule (whether similar or dissimilar), and the operations we can do with the plastic pieces (which may or may not reflect 'operations' possible in actual molecules). At a minimum, a model for model-based computation likewise consists of a set of representations and a set of operations upon these representations. The representations may be linguistic or symbolic, for example, while the operations may be thought of as functional relations between them.

[7] It is also interesting to note that a slide rule is typically called an analog computer. Under the misconception of analog computer as necessarily involving continuous variables, what role does continuity play in the use of a slide rule? It is arguable that the continuously adjustable aspect of the device is incidental to the actual use and function of the computer since outputs are also not real numbers.

3.1 Benefits?

In the present work I am remaining relatively qualitative in my analysis of the 'computational landscape', however some general comments may be of interest concerning any formal results associated with analogy machines or model-based computation. If there is any genuine 'speed-up' to be found compared to classical computation, I think it is primarily the result of two sources. The first potential source is simply due to the architecture and type of values processed in the given system.

The second, and likely more important, source of potential speed-up in any particular model-based computer is that it may front certain information in the 'premises' of the set-up.[8] In other words, some computational work might have already been done in the set-up of the system. This includes pruning off certain forks in reasoning or avoiding certain lengthy calculations that do not need to be investigated or reported by the program. As a simple example, just consider Deutsch's problem and finding out whether a black box implements a balanced or constant function of four possible functions $f : \{0, 1\} \rightarrow \{0, 1\}$.

A classical computer requires two evaluations of the black box, sending both a 0 and 1 through. We learn not only whether it is constant or balanced, but also *which* of the functions is performed. A quantum computer can, by throwing out the irrelevant information of the specific function and encoding the global property of the function cleverly into the phase, tell us in one go whether the box implements a constant or balanced function. Our model of the problem works with the architecture to cleverly set up the computation such that it (ideally) tells us only what we need to know and nothing more.

Any complexity claims should always be aware of these 'fronted' or indirectly utilized resources. If we haven't recognized these resources adequately, we might be mislead by certain claims of speed-up or complexity. In statements such as the following from Rubel, for example, we can see that these resources are alluded to by mentioning that the scientist has a 'feel' for the computing device:

> It is fashionable nowadays to downgrade analog computers, largely because of their unreliability and lack of high accuracy (roughly one-tenth of one percent at best). But analog computers, besides their versatility, are extremely fast at what they do, which is solving differential equations. In principle, they act instantaneously and in real time. Further, in contrast to the situation in digital computing, the operator of an analog computer has an extremely good "feel" for what the computer is doing. Analog computers are still unrivaled when a large number of closely related differential equations must be solved. (Rubel 1985, pp. 78–79)

While Rubel is specifically referring to analog computers, I think the statement is generally applicable to model-based computation. It is this 'feel' that I think imparts some of the benefits to model-based computation, since one has already done some work in constructing the model and in understanding how to work with the particular architecture. Many models provide the user with a 'feel' for the target problem or system, even with the acknowledgement that in reality there are certain features of the model which are non-representative or known to be false. See e.g. Frigg and Hartmann (2012, Sect. 4.2). By utilizing a model in computation, the features of the model (such as idealization, etc.) have restricted the computational possibilities to things which fit the use—thus streamlining any process to just those which are relevant for the User. For this reason, a model-based computer (or analog computer) is not necessarily a general purpose computer.[9]

4 Computational claims about the brain

We now move to an analysis of the first kind of computational claim to be discussed in this paper. In Searle (1990), Searle equates the question *Is the Brain a Digital Computer?* with *Are Brain Processes Computational?* After the preceding discussion, this seems like a mistake.[10] Digital computers are of course computational, but something that is computational is *not* necessarily *digital*. A digital computer may be repurposed in another context (with another user) to implement another form of computation, as noted by von Neumann:

> By an all-or-none organ we should rather mean one which fulfills the following two conditions. First, it functions in the all-or-none manner under certain suitable operating conditions. Second, these operating conditions are the ones under which it is normally used; they represent the functionally normal state of affairs within the large organism, of which it forms a part. Thus the important fact is not whether an organ has necessarily and under all conditions the all-or-none character—this is probably never the case—but

[8] The benefits or drawbacks of any specific device depends on the model involved, and thus it makes no sense to talk of 'general' speed-up (or slow-down) results for model-based computing.

[9] Piccinini (2015, p. 203) also notes that we should distinguish between 'general purpose' as referencing computational universality (i.e. a Universal Turing machine for digital computation), and 'general purpose' in the sense that we can do many things with the same device. A model-based computer in the account offered here might be neither universal nor useful for many different things.

[10] Even more so than Searle himself might have admitted.

rather whether in its proper context it functions primarily, and appears to be intended to function primarily, as an all-or-none organ. (Neumann 1963, p. 298)

To be fair, Searle's discussion does touch upon some very legitimate issues with these questions. However, it is not clear that his discussion translates easily for the notion of model-based computation advocated for here. I want to agree with Searle's (and von Neumann's) comments on *use* being fundamental to computation, but avoid the framing of computation as equivalent to *digital* computation. Digital computation is but one *subset* of potential user-dependent contexts which may constitute a computational device. Not only does a model-based notion of computation help clear this issue up, but it importantly emphasizes at its core the user-dependent context which is so central to the notion of computation generally. This helps us grasp better what alternative models of computation are doing for us, namely that they can be used to subjectively prune away irrelevance or to emphasize certain relevancies for particular uses.

Then, what would it mean if we asked *Is the Brain a Model-based Computer?*, and is this different still from Searle's second question *Are Brain Processes Computational?* In the scope of this present paper I cannot answer all of the interesting questions brought up in this topic, but I can discuss one recent approach in cognitive science that arguably fits the notion of model-based computer.

4.1 Bayesian brain and generative modeling

There is a growing use of Bayesian probabilistic methods and hierarchical generative models (HGM) in cognitive science. See e.g. Friston (2010) and Clark (2013). Some of this literature can be taken as arguing that the brain be considered a model-based computer as we have defined it here: that it is a device whose evolution in time effectively performs computations based on a *model*. Take Friston's description as an example:

The Bayesian brain hypothesis uses Bayesian probability theory to formulate perception as a constructive process based on internal or generative models. The underlying idea is that the brain has a model of the world that it tries to optimize using sensory inputs. [...] In this view, the brain is an inference machine that actively predicts and explains its sensations. (Friston 2010, p. 129)

This approach is argued by the authors to have the capacity of unifying several areas of cognitive science. Whether the specifically Bayesian approach is the final unifier may still be at question. Nonetheless, the approach not only fits the model-based account of computation I have advocated here, but even seems to fit the more restricted sense of analog computation since the modeling that a Bayesian brain is doing is related via similarity to the external world. The brain, under this view, is constantly simulating the world and adjusting its model according to the errors experienced. The HGM is amplifying relevant or similar features of a model via feedback with the environment, while dissimilar features fall out of focus (and, under the Bayesian approach, obtain lower probabilities).

Under this framework, we would answer 'yes' to the question of whether the brain is a model-based computer, and also 'yes' to the question of whether brain processes are computational. However, this may be a bit premature since we have noticed that computation is dependent on a user—and what would be *using* this model-based computer? This is no trivial problem, and in fact relates to longstanding mind/brain problems and what is called the "homunculus fallacy" (HF). See e.g. Searle (1990, Sect. V). Can the model-based conception of computation add anything new to this problem?

Without being overly conclusive, I suggest that the hierarchical generative model of cognition may be a good step towards addressing the user problem. The reason is that it simply accepts a finite regress and offers a more general notion of model-based computation.[11] While this isn't solving the problem (or avoiding the fallacy) in the traditional sense, it is simply not so unreasonable to suppose that the brain—as a computational system—involves complex hierarchical modeling of the external world. The representations in this model no doubt still succumb to HF objections, but not in a naive way.

The slightly more sophisticated view does not succumb to an infinite regress traditionally associated with the homunculus fallacy, since there are finite levels in the hierarchy. The homunculus could just be the topmost level in the hierarchical generative model, and it 'uses' the computations from lower levels in the hierarchy. Now, a reader familiar with the HF would likely object and say that the topmost level of the hierarchy is still problematic, since it is not 'used' by a higher level user. I do not know a way out of this objection, nor whether it is useful to reconcile. I can only say that the entire integrated 'body + HGM' system definitely seems to use the HGM, for all of the reasons why people think HGM is a good model of cognition in the first place.

In any case, it seems that a sophisticated model-based notion of computation does not do worse for computational

[11] Attempting to mitigate or explicitly accepting the HF is a required step, since as Searle notes, "...The homunculus fallacy is endemic to computational models of cognition and cannot be removed by the standard recursive decomposition arguments." Searle (1990, p. 36) What can be done, I argue, is to put a new spin on the issue.

claims about the brain than what has been accomplished previously. The brain doesn't need to be a digital computer, or a general purpose analog computer.[12] However, it is clear that a brain-like system which utilizes a model (i.e. does model-based computation) of the external environment to generate minimum error or minimum surprise is much different than Searle's formulation of these issues.

A potential benefit of this view of cognition was hinted at by Kenneth Craik in a 1943 publication:

> [I]n the particular case of our own nervous systems, the reason why I regard them as modelling the real process is that they permit trial of alternatives, in, e.g. bridge design, to proceed on a cheaper and smaller scale than if each bridge in turn were built and tried by sending a train over it, to see whether it was sufficiently strong. [...]
> It is likely then that the nervous system is in a fortunate position, as far as modelling physical processes is concerned, in that it has only to produce combinations of excited arcs, not physical objects; its 'answer' need only be a combination of consistent patterns of excitation—not a new object that is physically and chemically stable. [...]
> My hypothesis then is that thought models, or parallels, reality—that its essential feature is not 'the mind', 'the self', 'sense-data', nor propositions but symbolism, and that this symbolism is largely of the same kind as that which is familiar to us in mechanical devices which aid thought and calculation. (Craik 1943, pp. 52–57)

In cognitive science, this is now known as 'mental modelling', and it seems quite consistent with model-based computation as I have construed it here. Thagard characterizes mental models as "psychological representations that have the same relational structure as what they represent." Thagard (2010, p. 447) I argue that the benefits which Thagard and Craik attribute to the brain's ability to model are just the same as what I have outlined earlier. The low-level processes which contribute to model-formation end up trimming irrelevant representations off, and of course this is combined with parallelism in the neural architecture. Together with a system-defined *use* (i.e. the survival of a human or the optimization of some task), we can legitimately characterize 'mental models' in cognition as an instance of model-based computation. We can also find model-based computation in external scientific models, and in the next section I discuss a special case of MBC.

5 Analog simulation in physics

For our second kind of computational claim to be discussed, we move to physics. A few recent publications in the physical sciences (along with some philosophy of physics) have drawn attention to the use of analog models in scientific reasoning. One notable example is that of fluid systems displaying analogous phenomena to Hawking radiation (the phenomena of photons escaping the event horizon of a black hole). See Unruh (2008). These models have been argued, under strict conditions, to be performing analog *simulation* by Dardashti et al. (2015). Importantly, these systems seem to allow us more access to black hole phenomena than would otherwise be possible.

The reader should already be anticipating the main point of this section: these sort of systems are *analog computers* in the clearest sense—they are based on strict similarity conditions, and as alluded to earlier are prime examples of model-based computation (specifically analogy-based). They are simulating while also displaying formal and physical similarities with the target computational problem. The type of simulation these systems do is arguably providing even stronger results than traditional simulation in which the architecture of the computing device is irrelevant to the simulated problem. However, because of the background knowledge involved in constructing a table-top system, we might be less surprised at the outputs because we have a good 'feel' for what the system can do.

The strict models used in analog simulation are based on formal similarities (such as isomorphisms) between the systems of equations describing both the computing device (i.e. a table top fluid system) and the target system (i.e. a black hole). We mentioned earlier that for model-based simulation, the dynamics of the computation are relevant (but may not be similar). For analog simulation, the dynamics of the device must preserve relevant similarities with the dynamics of the target system. This brings us to the last important step in this short paper, namely re-connecting our discussion with previous work concerning model-based *reasoning*.

6 Model-based reasoning in science

The literature concerning model-based reasoning is divided among a few contexts. In philosophy of science, model-based reasoning is discussed in the sense of using a

[12] A similar point is made by Piccinini and Bahar (2013), however their re-structuring of the cognitive computationalism debate ends with a hybrid notion of computation for cognition that combines digital computation with the conflated notion of analog computation discussed earlier. Besides the obvious disagreement we have in defining analog computation, I think the notion of model-based computation discussed here supports in some ways their assertion that neural computation is a kind of its own (not particularly digital, not particularly continuous).

scientific model to make inferences about a target system (which the model represents). See e.g. Frigg and Hartmann (2012, Sect. 3). The systems discussed in the previous section are good examples. The scientist may, for example, use the model to justify a theory of Hawking radiation or to suggest new experimental questions. Inferences are carried from the domain of a model to assert information or knowledge about the target system. The model is assumed to have some relevant similarity to the target system.

There are, however, several other senses in which we might further distinguish flavors of model-based reasoning—particularly when talking about approaches to characterize human cognition. Consider a logical model, and the notion that a model-based reasoner chooses among sets of truth values of variables in a given premise set. That is, an agent's model is comprised of logico-semantic content, and model-based reasoning in this sense is some more or less straightforward deductive process.

This seems fairly distinct from the above notion, in that inferring from the domain of a model to a target domain seems not to be deductive, but rather abductive. The former notion of model-based reasoning has something to do with the seemingly non-deductive inferences that individuals and scientists make from a model that is not necessarily strictly 'logical'. For example, inferring some property of a real system from an idealized model in physics.

Another sense of model-based reasoning is in diagnostics, or in artificial intelligence systems which have a model of the environment. See Davis and Hamscher (1998). Now, this might sound very close to the notion of HGM and 'mental models' I discussed previously. However I think it is important to distinguish between model-based reasoning as somehow providing rules or guidelines in an argument or in an artificial inference system, with model-based computation as I have construed it. Model-based computation can be a part of model-based reasoning, but it isn't clear that model-based computation *is* model-based reasoning.

Reasoning is an active process (one might even say conscious), whereas computation—aside from the User's set up of a problem or the interpretation of an output—seems to be passively implemented. Model-based reasoning may likely be involved in constructing a particular computing device, but it isn't clear that what the device is doing should also be considered model-based reasoning. Or, it isn't clear that our *use* of the device as a model-based computer constitutes model-based reasoning as understood by previous work on the subject. Nonetheless, it seems to be that a more in-depth analysis of these two notions may be fruitful.

One could perhaps say that the most fundamental distinction to be made in this discussion is an *intra–inter* distinction. That is, there is the reasoning or processes involved in exploring a model—these inferences are *intra*-model. On the other hand, we have the seeming abduction or analogical argument from the model to the target system. This is *inter*-model, it exceeds the domain of the model and applies to our representation of the target system.

For a model-based computational device, it is important that our intra-model operations can produce a state which achieves inter-model significance. That is, at the end of the physical operations of the device, our *use* indicates a functional relationship of the outcome with some property of the target system (more precisely, a formal representation of the target system) or a solution of a target problem. In other words, the outcome corresponds under a 'use function' to a desired calculation. Under certain circumstances, there may be a formal similarity between the intra-model operations and the operations which could in principle be applied to the target system (or target problem). This would be identified with the special case of analog model computation and analog simulation.

7 Conclusion

At this stage, I offer a preliminary framework in Fig. 1 of computation to reflect model-based computation as discussed in this present work. I suggest that, at the very least, we should think of computation as consisting of two dimensions. One dimension represents the characteristics of physical devices and computational organs—or, our characterization of the dynamics or behavior of such components. The other dimension has to do with the status of representation in a model—ranging from a model which is totally dissimilar to a target problem, to very similar (like in analog simulation). Importantly, these dimensions can

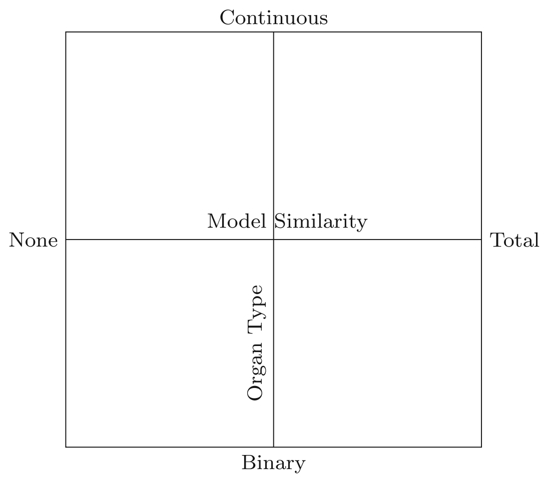

Fig. 1 A proposed picture of the computational landscape in the context of model-based computation discussed here

be thought of as more or less orthogonal, and we can see why not being conscious of these dimensions might lead to the kinds of objections discussed earlier concerning analog computation. Ulmann has drawn our attention to the fact that there are two fundamental dimensions of computation that need to be distinguished to really understand what certain devices are doing.

Briefly, lets take a look back at our examples and where they might fit into this framework. Take the slide rule example discussed earlier. The representations involved in the model do not particularly seem relevant or similar to multiplication. Even though our description of how the ruler implements multiplication is a step-by-step procedure, the manner in which the computation is achieved seems fairly dissimilar to a *particular* way of multiplying two integers. A slide rule is not the only means of computing multiplication, and so unless our target problem is explicitly to compute multiplication by a logarithmic means we would probably place a slide rule somewhere far to the left of *total* similarity.

The example concerning analog simulation, on the other hand, involves a table top model which is characterized by certain well-defined similarities (i.e. isomorphisms) to the target problem. This is very near to total similarity. The mathematical representations of the fluid system are similar to those of the black hole, as well as the dynamics ('operations') which occur when the system is allowed to evolve in time. Again, we can see that the more specific our model and device is, the generality of what can be computed is limited. It is unclear how multiplication could be accomplished by such a device, or why we would want to try.

Then there is the vertical dimension in the above framework. This concerns the types of values used by organs in the device. If only binary values are used, then this corresponds to a traditional digital computer. Somewhere in-between are any number of discrete-state devices. In the ideal case, we could imagine some device whose organs operated with continuous variables. There is always the question of whether such values are recoverable (i.e. quantum amplitudes), but nonetheless I think the extremities on this axis are reasonably clear.

This is all a preliminary outlook on what has so far been discussed. I have argued that this framework is beneficial for theoretical computer scientists and philosophers alike. Future work is justified, particularly in discussing more in depth case studies and attempting to plot them in such a landscape to get an overall picture of the landscape of computation. Even if the precise plotting of computational devices is not the primary goal, it seems like the exercise of discussing computation along these dimensions helps to get a grip on what one is working with in any given alternative computing device.

A model-based notion of computation helps us understand *why* certain architectures or models might perform better on, for example, optimization problems. Take D-Wave's supercooled annealing chip, for example. Its usefulness derives from a combination of architectural features and model-based considerations in the set-up of the device, and these determine the types of problems that it can be useful for. It is worth investing in because it exploits a combination of pre-computed modeling considerations with an architecture that also reflects these considerations (whether it truly exhibits a "quantum" advantage or not).

There is an interplay between our understanding and descriptions of physical systems and their dynamics, and what we consider to be our model for computation. For a general purpose digital computer, it might suffice to map a representational model (i.e. digital computation) onto our description of the physics of the device. On the other hand, it might be useful to first describe what a system *does*, and to see what could be computed if we set up and controlled such a system in certain ways. That is, let the description of the physics of the system define our model of computation. As an example, consider Carver Mead's outline of neuromorphic computing:

> The fact that we can build devices that implement the same basic operations as those the nervous system uses leads to the inevitable conclusion that we should be able to build entire systems based on the organizing principles used by the nervous system. I will refer to these systems generically as *neuromorphic systems*. We start by letting the device physics define our elementary operations. (Mead 1990, p. 1631)

The recent success of convolutional neural networks (CNN), as well as architectures such as IBM's TrueNorth, illustrate the model-based notion of computation very clearly for neuromorphic computing. A CNN is an artificial neural network, whose convolutional layers learn through back propagation appropriate kernels to convolve with the input. This is inspired by the receptive field of biological neuron vision processing. A convolution in the continuous case is defined for two functions (e.g. of time) f and g, sliding a kernel over a signal:

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\tau)g(t-\tau)d\tau \tag{1}$$

Even though TrueNorth was a neural-inspired architecture, it was not developed explicitly for deep CNNs. See Esser et al. (2016). The result of matching efficient neuromorphic architecture with neuro-comutational models is impressive. We see that as the model and architecture line up, the performance on certain classes of problems increases—for example, in machine learning and image recognition tasks. The tool fits better to the problem, encoding certain assumptions about what should be computed and why.

We can understand the justification to find an architecture that reflects our model as closely as possible, such as a more pure convolutional architecture. In the special case, we have a device that performs analog simulation as outlined earlier. However, this likely comes at a cost of computational generality. We may make some pragmatic compromises in our model for ease of use, for finding an appropriate architecture, and to increase the domain of application of the device.

In conclusion, model-based computation seems to be a worthwhile notion to entertain when discussing alternative computing. If a User wants to compute certain things that are modeled better by neuromorphic or quantum computers, then using these devices might provide some computational advantage. The notion of model-based computation does not entail any kind of dramatic proposal to re-draw complexity classes or endorse any view on hyper-computation. In fact, it is clear from the discussion that complexity claims should be wary of fronted resources by the modeling process.

As a conceptual tool, model-based computation helps us get a better grasp on the landscape of computation. This account also gives an intuitive picture of what does and doesn't compute (since a computational device computes relative to a User). I have argued that this tool is useful for analyzing and understanding various kinds of computational claims. Perhaps it can also help us keep track of the emerging market for specialized computing devices.

Acknowledgements I would like to thank Bernd Ulmann for introducing me to my own errors on the foundations of analog computation, and for motivating this present analysis. Among others, Michael Cuffaro, Ulrike Hahn, Stephan Hartmann, Christian Leibold, Gregory Wheeler, and Richard Whyman provided substantial feedback and discussion of the topic during the writing of this paper.

References

Beebe C (2016) Model-based computation. In: Amos M, Condon A (eds) Proceedings of unconventional computation and natural computation: 15th international conference, UCNC 2016, Manchester, UK, 11–15 July 2016, pp 75–86

Care C (2010) Technology for modelling: electrical analogies, engineering practice, and the development of analogue computing. Springer, Berlin

Clark A (2013) Whatever next? Predictive brains, situated agents, and the future of cognitive science. Behav Brain Sci 36(3):181–204

Craik K (1943) The nature of explanation. Cambridge University Press, Cambridge

Dardashti R, Thébaut K, Winsberg E (2015) Confirmation via analogue simulation: what dumb holes could tell us about gravity. Br J Philos Sci 68(1):55–89

Davis R, Hamscher W (1988) Model-based reasoning: troubleshooting. Memorandum AI Memo 1059, Artificial Intelligence Laboratory, Advanced Research Projects Agency, Office of Naval Research

Esser S, Merolla P, Arthur J, Cassidy A, Appuswamy R, Andreopoulos A, Berg D, McKinstry J, Melano T, Barch D, di Nolfo C, Datta P, Amir A, Taba B, Flickner M, Modha D (2016) Convolutional networks for fast, energy-efficient neuromorphic computing. In: Proceedings of the national academy of sciences

Frigg R, Hartmann S (2012) Models in science. Stanford Encyclopedia of Philosophy, Stanford

Friston K (2010) The free-energy principle: a unified brain theory? Nat Rev Neurosci 11(2):127–138

Mead C (1990) Neuromorphic electronic systems. Proc IEEE 78(10):1629–1636

Neumann JV (1963) The general and logical theory of automata. In: Taub AH (ed) John von Neumann collected works, vol V: design of computers, theory of automata and numerical analysis. Pergamon Press, Oxford, pp 288–326

Nielsen M, Chuang I (2010) Quantum computation and quantum information. Cambridge University Press, Cambridge

Piccinini G (2015) Physical computation: a mechanistic account. Oxford University Press, Oxford

Piccinini G, Bahar S (2013) Neural computation and the computational theory of cognition. Cogn Sci 34(3):453–488

Rubel LA (1985) The brain as an analog computer. J Theor Neurobiol 4:73–81

Searle JR (1990) Is the brain a digital computer? Proc Addresses Am Philos Assoc 64(3):21–37

Thagard P (2010) How brains make mental models. In: Magnani L et al (eds) Model-based reasoning in science and technology, vol 314. SCI, Springer, pp 447–461

Turing AM (1950) Computing machinery and intelligence. Mind 59(236):433–460

Ulmann B (2013) Analog computing. de Gruyter, Berlin

Unruh WG (2008) Dumb holes: analogues for black holes. Philos Trans R Soc A 366:2905–2913