HOW COMPUTATION EXPLAINS

Andrew Richmond

ABSTRACT

I discuss the monumental shift in our understanding of the brain triggered by the project of computational cognitive science: the use of tools, concepts, and strategies from the computer sciences to investigate the brain. Philosophers have typically understood this project, and the computational explanations it provides, to assume that the brain *is a computer*, in a sense to be specified by the metaphysics of computation. That metaphysics, by revealing what exactly we attribute to the brain when we say it computes, is supposed to show how and why computational explanations work, and in doing so to provide a philosophical foundation for them. In contrast, I give an account of computational explanation that focuses on the resources computational explanations bring to bear on the study of the brain. I argue that computational explanations help cognitive scientists build perspicuous models that capture precisely the kinds of causal structures they seek, and that no metaphysics of computation is required to understand how they do this.

1 Introduction

Cognitive science gives computational explanations of behavior. From neuroscience in particular we learn that the brain sees depth by computing the disparity between retinal images (Nityananda and Read 2017), discriminates colors using cone-opponent computations (Thoreson and Dacey 2019), localizes sounds by computing inter-aural time differences (Grothe et al. 2010), and supports reaching and grasping tasks by computing vector displacements (Shadmehr and Wise 2005). Cognitive scientists also make general appeals to the computational capacity of neural channels (Gallistel and King 2009), the computational architecture of the brain (Lake et al. 2017; Yamins and DiCarlo 2016), and the broad types of computation it can perform (Danks 2019).[1]

Perhaps this has all become commonplace enough to dull our critical instincts, but if we dwell for a moment on this fact, this explosion of computational explanations over the past half-century, some questions arise. What *are* these explanations? How do they work? What distinguishes them from other kinds of explanation? And why have they been so successful? (Not just successful, but so successful that it is hard to imagine cognitive science without them.) In its simplest form, the question is: *how* and *why* do computational explanations work?

[1]Discussions of the computational approach in general (as opposed to specific computational explanations) are also common (Cao 2019; Chirimuuta 2019; Fodor 1975; Gallistel and King 2009; Hardcastle 1996; Pylyshyn 1984, 1993).

The received view is that computational explanations work because the brain *is* a computer. The brain supports depth perception, e.g., by *literally computing* retinal disparity. If you accept this view, you will want to know what exactly it means — what it is to be a computer, and why so many successful explanations latch onto this property of the brain. So your inquiry has the form of a traditional metaphysical question. What is a computer? What features make something a computer? What criteria must something satisfy to be a computer? Call this the Metaphysical Approach to computational explanation.[2]

Note that the Metaphysical Approach doesn't want a theory of computation like the one Turing gave. That was a theory of computable functions, i.e. functions for which there exist effective methods or algorithms, and the nature of those algorithms. These functions and algorithms are formal, abstract things. But in the metaphysics of computation, we're not concerned with formal systems themselves. We want to know what it takes for a *physical system* to compute, i.e. to *implement* one of those formal systems. There are algorithms for addition, and then there are the cash registers that implement them. The question here is about the cash registers (and other physical systems): what does it take for a hunk of metal and plastic to implement an addition algorithm? What makes the cash register a computer, and what makes it the specific computer it is?

So, the Metaphysical Approach wants to identify the features of the brain that make it a computer, and that therefore make computational explanations of it appropriate and successful. But a satisfying metaphysics of computation is hard to come by. And when accounts of computation are unsatisfying, or when someone's metaphysics of computation suggests that any old rock (Putnam 1991), pail of water (Lycan 1981), or brick wall (Searle 1992) computes, one is liable to hear some familiar refrains. The brain isn't a computer; that's "just a metaphor." Or debates about what computation is and whether the brain satisfies that definition are "just semantics." A recent paper by Richards and Lillicrap (2022) exemplifies both frustrated responses: they argue that when we say the brain is a computer we either mean it's somehow like a laptop, which is just a metaphor (and not a very useful one), or we mean it implements a universal Turing machine, which is literally true but uninformative, reflecting only a stipulation about how we use the phrase "is a computer."

But even if the brain-as-computer-metaphor *is* a metaphor, it isn't *just* a metaphor — it's one of the most successful explanatory approaches in recent science. That needs explaining, and "it's just a metaphor" is hardle even a start. And even if they are semantic in nature, debates over what computation is aren't *just* semantics. As I've described, they are part of a broader project that aims to show how and why computational explanation works. "That's just how we use the word" is barely a first step in that project. We can dismiss the semantic debates if we like — in fact, I do. But unlike my fellow travelers, I take it that I'm not *just* setting aside those debates. I'm also taking up a burden: to explain the success of computational explanation in some other way. This paper is my attempt to discharge that burden. I think all the resources I need can be found in less controversial domains than the metaphysics of computation: I will appeal only to the kind of things computational notions allow us to do as we investigate and especially model the brain.

I'll illustrate the metaphysics of computation by introducing the *triviality problem* in section 2. In section 3 I'll turn away from the metaphysics of computation, and instead treat computational

[2]For representative examples, see Chalmers (2011); Shagrir (2022); Piccinini (2015). Things are no different if one thinks that computational explanations work because the brain *computes*, but is not *a computer*. The questions that arise are perfectly analogous (What is it to ~~be a computer~~ compute?), and the distinction won't be important for my purposes.

explanation as an example of the more general phenomenon of *domain transfer*: the use of tools, strategies, or concepts in a novel domain, or for a purpose they weren't originally developed for — like when companies apply NASA's failure-detection strategies to ad campaigns rather than rocket components (Edsel 2016), or when teachers use techniques from game design to make their courses more engaging (Miller 2014). I'll give an account of computational explanation that appeals only to *the resources it introduces into cognitive science*, and show that the metaphysics of computation, whatever it may be, is irrelevant to this account: as far as computational explanation is concerned, there might as well be no such thing as a computer. Call this the Pragmatic Approach. I'll develop the Pragmatic Approach in response to some objections in section 4, and conclude in section 5 with some final considerations that support it against the Metaphysical Approach. This will not just set the stage for further development of the Pragmatic Approach, but also clarify the terms of engagement with the Metaphysical Approach — the kind of argument for a metaphysics of computation that could, in principle, be compelling.

2 Triviality

In the examples I began with, the brain is not merely modeled computationally, like the weather often is (Ham et al. 2019). Weather models predict the future behavior of the weather, but often not the internal processes that bring it about. Genuine computational explanations, at least as they figure into cognitive science, do more than predict behavior. They are *process models*. They are supposed to explain a subject's capacities by telling us about the processes in the subject's brain that bring them about.[3] And, according to the Metaphysical Approach, computational explanations tell us that the brain brings those capacities about *by computing*, or by *being a computer*.

The burden is to say what exactly this means. What is it to be a computer? What criteria do we apply to tell whether something is computing, or what it is computing? The main hurdle for philosophers answering these questions is the *triviality problem*. Many seemingly plausible answers to these questions end up counting too many systems as computers, and counting any given system as computing too many things. And on the Metaphysical Approach, this spells disaster for computational explanation. This section will describe the triviality problem in more detail, as a way of illustrating the Metaphysical Approach and its differences from the Pragmatic Approach.

It is standard to introduce triviality using the simple mapping account of computation (Egan 2014), according to which a system implements a computation if its states mirror the stages of the computation, i.e. if there is a mapping between the stages of the computation and the system's states. On this account your calculator computes addition because when you punch "5" and then "7", the display shows you "12," and in doing so it has transitioned from states that map to the numbers 5 and 7 to a state (the output) that maps to the number 12. To compute some more detailed algorithm, a system must only transition between physical states in a way that preserves a mapping to the algorithm's more numerous stages. We can set aside some niceties of definition and say that *a system performs a computation just whenever its physical dynamics map to the dynamics of the computation*.

[3]This is not the only use to which computational models are put. They can be more than predictive models, but still less than process models. E.g., they can specify the optimal functioning of a system (Sánchez n.d.), or explain why it is the way it is (Chirimuuta 2014). The point is well taken, but these uses of computational explanation are not my target here.

This is intuitive. Ask a computer scientist what makes something a computer and they'll likely give you the mapping account. But it has a problem: mappings are cheap. Virtually every system maps to virtually every algorithm or computation, given a suitable 'carving up' of the system in question. For instance, if we want a rock to compute the addition function, we need only decide which instances of addition we'd like it to perform in a span of time. If we'd like it to have just now computed the addition function for inputs 5 and 7, we take the past three seconds of the rock's existence and map its state at each second to one of the numbers: it transitioned from state-at-second-1 (mapped to 5) and state-at-second-2 (mapped to 7) to state-at-second-3 (mapped to 12).[4] There is nothing special about the rock. This is true of every object that has been in at least three states over the past three seconds. And it's not limited to simple computations. If we want the rock to compute the addition function using, say, the same algorithm your calculator does, we map its physical states over a span of time to the stages of the algorithm your calculator follows.[5] According to the simple mapping account, this would show that the rock computes addition just as your calculator does. This is treated more rigorously by Putnam (1991) and Chalmers (2011), but the upshot is simple: mapping relations are too numerous to constitute a metaphysics of computation, because they make it too easy for a physical system to be a computer. And that saps or renders mysterious the explanatory force of computation in at least two ways. I'll dwell on them for a moment, because they nicely illustrate the contrast between the Metaphysical Approach and my own.

First, much debate in cognitive science is over which computations the brain performs.[6] It is because the brain performs cone-opponent computations and not simple cone summations that it supports the kind of color vision it does (Jacobs 2014). If it performed both computations, the connection between both models and their explananda would be severed: cone-opponent models couldn't make a prediction about color vision that cone-summation models didn't also make, and vice versa, because each would have to allow that the brain also performed the other model's computations. It is because the brain performs certain computations *and not others* that those computations explain its capacities. Consider also the *discovery* of the brain's computational properties, which the mapping account renders far easier (just find a mapping — as easy to do with the brain as with a rock) than the history of cognitive science would suggest.

And second, consider systems other than the brain. Even if we find sufficiently narrow criteria so that the brain computes only a limited set of functions or algorithms, it is a problem if too many other things also compute them. If it turns out that a rock implements an addition algorithm, then your brain's implementing that same algorithm could not explain its arithmetical performance, because that algorithm can be implemented without supporting arithmetic. Performing a computation needn't be *sufficient* for addition — other background features may be involved. But if those background features are computational (e.g., the computations the brain performs to use the outputs of the arithmetic module), the same problem arises: the rock will have them too, on the simple mapping view. And if the background features are not computational, then non-computational properties do

[4]Repeating states — e.g., if the rock's next calculation includes a 5 as well — are handled by disjunctions, so it is state-at-second-1-or-4 that gets mapped to the number 5.

[5]And of course if we carve up the rock's states more finely, we can map it to complex computational structures like whole Turing machines or neural networks.

[6]We might say the brain is performing every computation, but only some are explanatorily relevant. But this just pushes the question back a step: we have to say what makes a computation explanatorily relevant, and that framing doesn't seem to offer any additional traction on the issue at hand.

all the work making the difference between a system capable of arithmetic and a system incapable of it; computation is irrelevant, and again we've undermined its explanatory role.

The problem, then, is that rather than illuminating computational explanation, or providing it with philosophical foundations, the mapping account appears to *undermine* computational explanation and make its success mysterious. If we're approaching computational explanation through the metaphysics of computation, these problems have to be solved by an appropriately narrow definition of computation: one that limits which systems implement which computations, in a way that preserves the scientific role of the notion of computation. E.g., we might consider the causal view (Chalmers 1996), which requires an algorithm to map to a certain kind of causal structure in a system. Other attempts have grounded computation in not just the causal but the teleological (Milkowski 2013; Piccinini 2015) or representational (Peacocke 1994; Shagrir 2018) properties of computing systems.[7]

It hardly needs to be said that these views are all controversial. It is debatable whether the causal view alone solves our problem. Scheutz (2012) and Shagrir (2001) argue that the causal view still allows a problematic proliferation of computations, and Egan (2012) argues convincingly that even if it is safe from triviality, Chalmers' definition of computation is not suitable for the use cognitive scientists put the notion to.[8] And teleological properties — properties to do with something's *function* or *purpose* — incur severe explanatory debts themselves, so it's not clear they put the metaphysics of computation on better footing (cf. Dewhurst 2018).[9] Of course there are similar, if less severe challenges for most views of representation as well (e.g., see Egan 2019).[10] I note these issues not as an argument against the causal, representational, or teleological views of computation, but only to make clear the problem that the Metaphysical Approach revolves around: defining physical computation so as to include all the right systems and exclude all the wrong ones. This is supposed, on the Metaphysical Approach, to be the first step in explaining how computational explanation works.

What is distinctive about the Pragmatic Approach is that the triviality problem and all of the resulting puzzles will be irrelevant to it. The main desideratum for the Pragmatic Approach is the same as for the Metaphysical Approach: to explain how and why computational explanation works. But the Pragmatic Approach does not achieve this via the metaphysics of computation — instead, it looks to the goals computational explanation serves and how it serves them. If this approach successfully explains computational explanation, it will have shown that the metaphysics of computation is unnecessary: a careful look at how computational notions work, and what they do for cognitive scientific explanations, is enough. This is what I'll try to show in the next section.

[7]It is worth noting that these views are generally *elaborations* of the mapping account. They accept the necessity of a mapping, but add other conditions that must be satisfied for a physical system to compute. This is also a plausible reading of Egan's own version of the mapping account, where the additional conditions have to do with features of our explanatory context (Egan, in conversation). In that case my own view will be consistent with the mapping account, and in fact the two will fit nicely together.

[8]The causal view also appears unable to account for the apparent environmental individuation of computational processes (Shagrir 2001; Richmond n.d.*a*; Shea 2013).

[9]See Chalmers (2011, 334) and Sprevak (2019, 177), on the need for computation to be grounded in well-understood notions.

[10]Doubly so for the most popular view of representation, which grounds it in teleological properties (e.g., Neander 2017). But representation can still be part of the story, and Richmond (n.d.*b*) shows how the Pragmatic Approach can preserve much of what is most important about the representational view.

3 Domain transfer and a pragmatic approach to computational explanation

The goal of this section is to build an account of computational explanation in the space between two extremes. First, the account shouldn't pull us into debates about the metaphysics of computation, or about whether the brain *really is* a computer. But, second, it should explain the ubiquity and success of explanations that conceive of the brain in computational terms — it should say more than "it's just a metaphor."

3.1 Making room for the pragmatic approach

First, let me reiterate something from the previous section. I distinguished between merely predictive computational models and genuine computational explanations. One way of making this distinction more precise is to say that computational explanations give *process models* of their target systems — models of the processes that generate their behavior. Simon and Newell expressed this early on:

> We do not say that we understand the magic [trick] because we can predict that a rabbit will emerge from the hat when the magician reaches into it. We want to know how it was done — how the rabbit got there. Programs like LT [the authors' "Logic Theorist"] are explanations of human problem-solving behavior *only to the extent that the processes they use to discover solutions are the same as the human processes*.[11] (Simon and Newell 1973, 147, my italics)

So there is the distinction between merely predictive models, like computational models of the weather, and process models, which detail the processes a system undergoes to generate its outputs. But process models are not necessarily models that attribute, to their target systems, membership in a special category. Consider models of physical bodies expressed in calculus equations. These models can be more than predictive — they can model the processes that bodies go through to generate their dynamics or final positions. But though the model is couched in calculus, the system it models need not be a *calculizer*, or meet some criteria for membership in that category. We don't think the model unveils any *inherent calculus-properties* of the system. Thinking that way about *computational* process models in particular would be a sharp divergence from our treatment of most scientific modeling. For an even starker example, consider models of fluid dynamics applied to traffic jams (Sun et al. 2011) or epidemiological models of disinformation (Kucharski 2016). These models don't require or assume that traffic literally is a fluid, or that disinformation is, by some criteria, a virus.

These are just examples of domain transfer: tools, concepts, or strategies that were developed for one purpose or domain are being applied to another. Hospital teams have borrowed strategies from Formula 1 pit crews and dance choreographers to hand off patients from surgery to the ICU (Sower et al. 2008). They are not performing a ballet, and their patients are not assumed to meet the criteria for the category FERRARI or MCLAREN. Nor do we need a metaphysics of race-cars to understand how and why the hospitals' strategies work. All we need is to understand how the tools from one domain work, and why they work in another. And, at the risk of belaboring the point, these questions

[11]Marr (1982, 23) expresses a similar sentiment. Also see Fodor (1968), Kriegeskorte and Douglas (2018), Sun (2008), and Nancy Dice in Bailer-Jones (2002) on computational models as process models.

need not be answered by saying that the elements of the two domains share a status as viruses, fluids, or race-cars.[12] So it is premature to say, as Chalmers does, that "[w]e cannot justify the foundational role of computation [in cognitive science] without first answering the question: *What are the conditions under which a physical system implements a given computation?*" (Chalmers 2011, 325). We'll know whether we need to answer that question only once we understand what we're doing when we give computational explanations, and how this approach serves cognitive science. If the approach works like a typical domain transfer, the project Chalmers describes is unnecessary. In section 3.2 I'll argue that computational explanation *does* work like a typical domain transfer.

3.2 How computational explanation serves cognitive science

Cognitive science wants to explain the cognitive capacities of complex systems like biological organisms: the capacity to detect and distinguish between stimuli, to decide on a course of action, to navigate spatial and social environments, and so on.[13] These are capacities to produce appropriate behavior in a range of environments and given a range of inputs. And cognitive science, since the rejection of behaviorism, aims to explain these capacities by appeal to the internal causal structure of the system in question. It is this structure that mediates the system's behavior; it is this structure that determines its solutions to the problems it faces; and it is this structure in terms of which we understand that behavior and those solutions.[14]

These goals call for a description, at an appropriate level of detail, of the brain's causal organization and processes, along with conceptual resources to *explain* behavior under that description. This means we need, at least:

(A) A language or formalism with which to describe the causal structures in the brain that support cognitive capacities.

(B) Conceptual resources with which to form questions, hypotheses, and explanations regarding those causal structures and the way they support cognitive capacities.

(C) Heuristics and background knowledge that make it efficient to form and work with these hypotheses and explanations.

The need for expressive languages and formalisms, as in (A), is widely discussed. See Lazebnik (2004), e.g., on the importance of a good formalism in biology for making predictions, framing

[12]They must share something for the domain transfer to be successful — something about the two domains must explain why the same tools can be used in both cases. The point is that it would be silly to think this something was their status as race-cars, rather than that pit crews working on a car do so with certain goals and under certain constraints, and that the goals and constraints for the hospital team are comparable in some respect.

[13]The following also holds for less traditionally "cognitive" capacities, like emotion regulation.

[14]A caveat: we do not want the most detailed or accurate model of that causal structure. We want a model that coarse-grains, idealizes, and is occasionally outright wrong, wherever those features are theoretically fruitful. We model a calculator with an addition-function, not an addition-except-where-the-calculator-errs function. So in the case of computational explanation, like causal modeling in general, accuracy with respect to causal structure is just one goal, tempered by others. I'll set this aside for now, but see Richmond (n.d.*b*) for some detail about how representational thinking supports this idealization and coarse-graining.

hypotheses, revealing important features of the target system and making them salient, and providing unity to the field. But it is already implicit in this that not just any language, even a highly descriptive one, will do, and (B) is required to ensure that our formalisms are appropriate to our subject matter — together with our conceptual resources, they should facilitate *theoretically useful descriptions* of the causal structures we seek (Lazebnik 2004). Cognitive science does not just need a language to *describe* the brain, but also to state its explananda and to frame relevant questions and hypotheses about those explananda. A formalism borrowed from particle physics might do a good job of describing the structure of the brain in many respects, but it would likely be difficult or impossible, within that formalism, to state explananda having to do with (say) an organism's capacity to memorize strings of words, or to frame hypotheses about the aspects of the brain's causal structure that allowed it to (say) reason deductively. A formalism that failed in these respects would need to be either abandoned or supplemented with conceptual resources that allowed it to do this work. These two desiderata, (A) and (B), will play the largest role in my discussion. (C), although it is an important part of any research program, is more nebulous. Among other things, (C) points out that the descriptions, predictions, explanations, and models that (A) and (B) allow us to give should be cognitively tractable for scientists, facilitate further investigation, have well-understood or clear properties, and the like. Together, what (A)–(C) would give us is a set of tools with which we can describe the causal structures in the brain that bring about its behavior and support its cognitive capacities (from A), understand those structures in relation to our explananda and form relevant questions and hypotheses about them (from B), and work efficiently on those questions and hypotheses (from C).[15] (In addition to these desiderata, it is also crucial that our formalisms and conceptual resources make it possible to *test* our theories. But this is essentially the problem of giving them empirical content, and I'll treat that in section 4.)

My claim is that computational notions provide a set of formalisms and conceptual resources satisfying (A)–(C), and as such they contribute to the goals of cognitive science by providing resources to explain how the internal structure of a system brings about its cognitive capacities. Since they do this well, they facilitate good explanations. And since those explanations hinge on fruitful and relevant descriptions of causal structures — not on the subsumption of a target system under the definition of *computer* or *computes* — then we need not worry about that definition and whether the brain satisfies it, any more than we worry about the definitions of *calculizer*, *virus*, *fluid*, or *race-car* in the earlier examples. The task, then, is to see how (A)–(C) are satisfied by computational notions and formalisms.

I'll start by focusing on formalisms. We need a formalism in which to capture the kinds of causal structures cognitive scientists seek — causal structures that explain cognitive capacities. There is no unique formalism appropriate to this task, and, for that matter, it is unclear how formalisms should be individuated. In particular, what counts as a computational formalism is not straightforward, and seems to depend on how tools from computer science are exported into new domains (Smith 1999). The formalism of Turing machines is used in computational explanations, as are the formalisms of finite state automata and combinatorial state automata, the formalisms involved in describing perceptrons and artificial neural networks, on through more generic forms of description like wiring diagrams (e.g. Sejnowski et al. 1988), arithmetical operations (e.g. Devalois and Devalois 1993),

[15] I'm not claiming this is all that a formalism and its associated conceptual resources should do for a field of science. But (A)–(C) alone will serve to display much of the way computational explanations function and the reasons they succeed.

calculus equations (e.g. Shadmehr and Wise 2005), and statistical functions (e.g., when a neuron is described as computing a Laplacian of Gaussian function, Egan 1999, 192).[16] So I won't rigorously define "computational formalism." Instead I will lean on the way the notion of computation is used in cognitive science, and I'll allow that whatever, for cognitive scientists, counts as a computational description, is a computational description. The task is to see what those descriptions have in common that makes them suited to achieving the goals I've outlined.

What the formalisms above have mostly in common is that they invoke the devices built by computer engineers (e.g., wiring diagrams), the programs designed by computer programmers (e.g., neural networks), or the mathematical structures investigated in computer science (e.g., Turing machines and finite state automata). In using these formalisms, cognitive science describes the brain in terms borrowed from the science, engineering, or programming of computers, broadly construed. I'll condense this by saying it uses formalisms borrowed from the computing disciplines. These formalisms — call them the computational formalisms — are computational explanation's answer to (A). The conceptual resources that allow computational explanation to meet (B) and (C) are the ones attendant on the relevant formalisms, or that are otherwise drawn from the computing disciplines.[17]

The case to be made is that these formalisms and conceptual resources serve (A)–(C) well: that describing and conceptualizing the brain using them serves cognitive science's broader goals. So I will turn now to some of the ways that computational formalisms and their attendant conceptual resources serve those goals. One important feature of computational formalisms is their facility with functional abstraction. Functional abstraction highlights an aspect of a system component, usually described mathematically, that captures its contribution to the system's behavior at a higher level of abstraction. A paradigmatic example is naming high electron flow in a wire "1", and low electron flow "0" (Hillis 1998, 18-19). Any variation in the electron flow within either "1"-signals or "0"-signals disappears, along with the gradient between "1"- and "0"-signals, and all the wire's other features. In fact, the wire itself disappears. All that remains is the distinction we've selected as significant for our purposes — a distinction between 1 and 0. Because we've chosen a description of the wire under which it behaves predictably (we know the circumstances that will put the wire in a 1-state and the circumstances that will put it in a 0-state), we can exploit that distinction to build more complex functions like logic gates.[18]

There is no need to belabor the utility of functional abstraction for engineering, but it is important that it offers benefits in the reverse-engineering of the brain as well, particularly in a computational context. The saltatory action potential, e.g., lends itself well to a characterization in terms of 1s and 0s; this was an explicit motivation for von Neumann's (1958) and McCulloch and Pitts' (1943) treatment of the brain in computational terms.

The story is now, of course, much more complicated. We don't treat neurons as logic gates but as (something at least as complex as) non-linear functions of weighted sums of inputs. But

[16]And to the extent that artificial intelligence informs cognitive science, its developments will introduce new and unpredictable computational formalisms (Kriegeskorte and Douglas 2018). The concept of computation, as it figures into cognitive science, is 'open-textured' (Waismann 1968) in at least this respect.

[17]I'll return, in the next section, to borderline cases that may not be captured by my definition, like the arithmetical operations mentioned above.

[18]The examples here are simplistic for the sake of explanation. Functional abstraction is most commonly discussed in more demanding contexts, e.g. abstraction methods for managing complex databases.

it is still a major goal of cognitive science to "decompose cognition into functional components," and to discover how the brain's activity at an "elementary" or neural level can be characterized so as to compose those functions (Kriegeskorte and Douglas 2018). And to do this we need a way of abstracting from the complex causal profile of neurons (or ensembles of them, or brain areas) to well-understood mathematical functions. To be clear, functional abstraction is an unavoidable feature of the mathematical description of any physical system. The question is which mathematical formalisms to use. And what better formalisms than the ones for which we understand the implications most relevant to us? We know a great deal about the processes defined by computational formalisms: how fast they are, how many steps they take (if they are step-wise processes), how they scale to different inputs, how efficient they can be at what cost to accuracy, what they can do with and without recurrent steps, and so on (e.g. Kriegeskorte and Douglas 2018, Box 3), and these are many of the same questions we have about the brain. So computational formalisms give us a toolkit for functional abstraction that is particularly well-suited to the questions we have about the brain.

Computational formalisms and the conceptual frameworks they bring with them also lend themselves to descriptions in terms of *algorithms* and *hierarchies*. Algorithms are functions strung together into (formally computable) sequences. They provide a clear and intuitive way of connecting a system's inputs to its outputs by describing the steps taken by the system in transforming inputs to outputs. That kind of description is precisely what cognitive scientists seek, as I suggested above: a description of the internal causal sequences that bring about cognitive capacities, the latter understood in terms of responses (or outputs) to environmental conditions and stimuli (or inputs). E.g., color-processing in early vision is modeled as an algorithm first summing responses from different types of cone, then weighting those sums, then adding and subtracting the weighted values, and eventually plotting the results in a three-dimensional space (Devalois and Devalois 1993; Mancuso et al. 2010). That is a description, in terms of an algorithm, of the way the brain turns a retinal input into a behavioral (or phenomenal) output.

Hierarchies are processes that operate at more than one level of abstraction. One kind of hierarchical description is just an important kind of algorithmic description. Neural network models show us how a process can derive more and more abstract or high-level features of an input, e.g. an image, through a series of functions that finds its low-level features like lines and shapes, then intermediate-level features, and eventually high-level ones like object types (e.g. *dog* or *cat*). This was also an explicit goal of earlier, classical computational modeling (Marr 2010). In computational neuroscience, this kind of hierarchical algorithmic description is essential to understanding how the brain makes categorizations, and especially how it proceeds from sensory stimulation to sophisticated high-level categorizations and behavior based on them. Computational frameworks drawn from neural networks (and, earlier on, other sources in the computing disciplines, Marr 2010) have helped illuminate the relevant brain processes by providing useful and increasingly accurate models of them (Richards et al. 2019).

Another kind of hierarchical description is *compositional*, rather than algorithmic. A compositional hierarchy is not a series of functions deriving higher- and higher-level features, but a hierarchy where a small set of simple functions compose more and more complex ones. E.g., the processes involved in different capacities may rely, at a lower level of their hierarchies, on a small "set of standard (canonical) neural computations: combined and repeated across brain regions and modalities to apply similar operations to different problems" (Carandini 2012) (see also

Carandini and Heeger 2012). An understanding of this sort of hierarchy does a number of things for cognitive science. Understanding the simplest neural functions guides anatomical investigation into basic units and circuits and makes salient certain aspects of their causal structure (Carandini and Heeger 2012). Without the simplicity and structure given by hierarchical thinking, it would be prohibitively difficult to connect cognitive neuroscience to basic physiology and anatomy, or generally to lower levels of brain organization. An understanding of how low-level brain structures compose high-level ones also benefits modeling, since it reveals relevant and practical levels of description, particularly when the goal is (as is common) to model how high-level behavior results from low-level organization (Yamins and DiCarlo 2016). For all these purposes, the benefits of computational formalisms are clear: their hierarchical properties are relatively well-understood; many computational formalisms are developed precisely for their ability to compose complex functions from less complex ones, especially a *small set* of less complex ones (particularly relevant when we consider canonical computations as above); and they are developed to create and make intelligible complex relationships at different levels of detail and abstraction.

The importance of hierarchical and algorithmic explanation, and the way computational formalisms accommodate them, is the last point I'll raise in support of computational formalisms being a good solution to (A). Moving on to (B), the assimilation of one system under the conceptual scheme developed for another system is a widespread and natural part of science (Dunbar 2002; Nersessian 2002), and conceptual schemes from the computing disciplines have been particularly useful ones in which to assimilate the brain. In fact, the discussion so far has already shown that computational formalisms and the conceptual frameworks attendant on them are well-suited to frame explananda to do with cognitive capacities, and to pose questions and hypotheses about the processes that bring them about. E.g., I discussed the way neural networks provide frameworks for thinking about how the brain's causal structure supports the derivation of object categories from lower-level features of a stimulus. But it is worth noting a few more cases. E.g., considerations of algorithmic complexity — an important concept in computer science — drive discussions about the appropriateness of Bayesian models of the brain (Kwisthout and van Rooij 2020). Considerations of computational efficiency — important in computer science and computer engineering — drove early debates in cognitive neuroscience (McClelland et al. 1986), and considerations of "computational cost" drive current discussions of navigation and route planning (Daniel et al. 2015). It is because we think of the brain in computational terms that we investigate its canonical operations, as above. It is because we understand the properties of recurrent connections in neural networks that we look for recurrent connections among neurons (Richards et al. 2019). The search for the brain's learning rules and functional architecture is spurred and supported by thinking of it in terms explicitly borrowed from the study of neural networks (Richards et al. 2019). It's clear that assimilation into the conceptual schemes of the computing disciplines provides many concrete benefits for generating hypotheses, and for understanding our causal models of the brain in relation to their explananda. If this seems to belabor the obvious, recall that the take-away is not that it is useful to think of the brain as a computer — that *is* obvious. The take-away is that we can make sense of this fact without claiming that the brain *is* a computer, and without entering into vexed questions about what exactly that means. All we need is to understand how computational formalisms are used to describe causal structures, and why they are so useful for this task, given cognitive science's particular goals, constraints, and explananda. That's what I've described above, with no need for metaphysical commitments; in their place I have appealed only to scientifically common-place considerations about explanatory goals and the tools with which we pursue them.

To finish with a familiar point about (C), computational explanation makes it possible, and relatively easy, to *build* models. Compare Kriegeskorte and Douglas (2018): "only synthesis in a computer simulation can reveal what the interaction of the proposed component mechanisms [of some theory] actually entails and whether it can account for the cognitive function in question." It is a common refrain in the history of cognitive science that computational models make hypotheses clear and specific, and they do this partly by making them buildable using our current technology (e.g. Churchland and Grush 1999; Pylyshyn 1984; Samuels 2019; Sejnowski et al. 1988). To grasp the significance of this one need only imagine a theory of the visual cortex as a convolutional neural network, but imagine it proposed 50 years ago. The benefits this theory has because of current computing technology (to do with prediction, ease of understanding, the availability of proofs of concept, our familiarity with the model and an intuitive understanding of what it says about its target system, etc.) are the benefits I'm claiming computational models have in general because of their buildability, and because of the familiarity we therefore have with them.

That's all I'll say about (A)–(C). To summarize, computational explanations, by drawing formalisms and conceptual resources from the computing disciplines, support cognitive science by providing: explanatorily relevant functional abstractions of the brain's causal structure; descriptions of that causal structure in the fruitful terms of algorithms and hierarchies; tight connections between our descriptions of the brain's causal structure and cognitive science's *questions* about that causal structure; relevant and fruitful ways of framing answers or hypotheses concerning those questions; and, more generally, explanations of how the brain supports cognition that are natural, powerful, and sensitive to our specific interests in cognition and our existing knowledge of the brain. And it does all of this with relative efficiency by drawing on well-understood/understandable, well-established, and deeply-ingrained conceptual frameworks. There is more to be said about the details here, but it is a benefit of the discussion so far that it relies only on uncontroversial features of computational formalisms and concepts. It is not that *on a certain tendentious way of thinking about computation* we can do without a metaphysics of computation. To make progress on our questions about computational explanation, and to do so without entering into the metaphysics of computation, we need only appeal to the features of computation that we are all aware of.

The goal was to sketch a view of how computational explanation works and why it is so successful in cognitive science. The view, more succinctly, is this: computational explanation works by using computational formalisms and the conceptual resources attendant on them to construct process models that capture the causal structures in the brain that bring about its cognitive capacities. If this is what computational explanation does, it requires no assumption that the brain is a computer, much less a theory about what it is to be a computer. This account reveals, at a general level, how computational explanation works, and also why it is so successful: it serves the purposes of cognitive science exceptionally well. The account also shows us what makes computational explanation *distinctive* as a mode of explanation — not the subsumption of the brain under a special definition or category, but the powerful suite of tools, resources, and concepts it draws on to serve the particular purposes of cognitive science.

4 Objections

The bulk of the Pragmatic Approach is on the table. In this section I'll consider some objections, focusing on ones that will let me sharpen the approach a little further. To bring out the most pressing worry, let's start by returning to a version of the triviality problem.

4.1 Empirical content: another triviality problem?

The triviality problem can't arise in its original guise: it attacked the definition of 'computer,' or the criteria for membership in the category COMPUTER, which are not involved in my view. But it might be resuscitated along the following lines: I've described computational explanation as a certain kind of modeling practice, but I haven't said how to tell what a given computational explanation concretely says about its target system. That is, I haven't yet placed any constraints on the *empirical content* of computational explanations. And if there are no constraints on the empirical content of a model, then why can't we interpret it as saying whatever we like? Why can't I give a computational process model of a rock as performing addition, and say that the model is correct as long as the rock runs through time-slices that correspond to the stages of the model's addition algorithm? This was a long way to come, just to end up back at the original problem.

But the problem is only apparent. Computational explanations say something about the causal structure of their target systems, and two constraints ensure that what they say about this structure is non-arbitrary. First, this version of the triviality problem is a special case of a more general problem: in virtue of what does any model say what it says about the system it says it about? I don't propose to answer this question here, but on the view I've defended, the question of the content of a computational explanation is just an instance of this more general question of model reference (Frigg and Nguyen 2018; Frigg and Hartmann 2018). And we can be sure that it is not arbitrary what scientific models in general (and therefore computational explanations in particular) say about their target systems. Whatever your account of scientific models, you have to explain their empirical content somehow, and there is no reason that computational explanation in particular would be excluded by whatever account you adopt. Note also that the problem of model reference is not generally solved by criteria for a target system's membership in a particular category. The revision of the triviality problem I'm considering would apply equally well to models of the solar system constructed in calculus equations, and the problem of why those models say what they say about their target systems is not solved by criteria for being a *calculizer*. Likewise for models of traffic from fluid dynamics, models of disinformation from virology, and so on. So the problem of model reference is no argument against the Pragmatic Approach. If you think models can have non-trivial empirical content, there is no special reason to worry about computational explanations.[19]

The first constraint, then, is the general theory of model reference — whatever we say about *that* will apply to and constrain the content of computational models. The second constraint comes from existing scientific knowledge. The goal of a computational explanation is to describe the causal structure that brings about a system's capacities. Not just any empirical content is appropriate for

[19]Others have made similar points. Matthews and Dresner (2017) argue that triviality arguments about computation have the same structure as triviality arguments about any attribution of numerical properties to physical systems, and so cannot hold. The advantage of the Pragmatic Approach is that it says something positive about computational explanations and their success, and shows what exactly is wrong with the triviality problem: it challenges computational explanation only under the Metaphysical Approach, which is — I've suggested — mistaken.

this task. If a neuroscientist held that her model of memory was confirmed by connectome data because that data revealed that the brain had *just some mapping* to her model, she would be laughed out of the lab meeting. But many more specific mappings would also be dismissed. What counts as an appropriate mapping of model to brain (appropriate empirical content) depends on background neuroscientific knowledge about (e.g.) which aspects of brain activity are involved in the tasks she's modeling, which components of the brain are causally efficacious in the right ways, and so on. To explain a memory task, she might propose that synaptic weights correspond to certain terms in her computational model. This would be appropriate if synaptic weights were causally implicated in memory in the required way, but it would be inappropriate if Gallistel and King (2009) were right that synaptic weights cannot bear significant responsibility for memory. Empirical constraints on model interpretation are familiar to cognitive science — e.g., see discussions of "mappable" models (Yamins and DiCarlo 2016) or "explanatory mechanisms" in model building (Blohm et al. 2020). For more concrete examples, consider debates over whether neural spike *rates* or *timings* are causally efficacious in the brain, and which should be the target of our models (Brette 2015). Or see debates about modeling population activity vs modeling individual neurons and their connections (Barack and Krakauer 2021). The empirical content of computational models is partly constrained by the diverse empirical considerations that constrain the empirical content of all models, as we should expect.

4.2 Applying computational explanations

There seem to be circumstances where computational explanations are appropriate, and ones where they aren't. It might be objected that because I've set aside the category COMPUTER, I can't say which systems should receive computational explanations (computers) and which shouldn't (non-computers).

So, why is it not always acceptable to use computational explanations, regardless of one's target system or explananda? In one sense, it is! We should have no qualms with someone to whom computational formalisms and conceptual resources derived from the computing disciplines are helpful for explaining (say) planetary systems or the weather, because accommodating this does not require a revisionary metaphysics according to which planetary systems and weather patterns are computers. We should note, however, that computational explanations are in fact not helpful in most cases, at least to us in our current context. As I've described computational explanation, it introduces a particular set of resources that are useful for a particular set of goals, in a particular scientific context, given particular constraints imposed by our target systems and the nature of our inquiry. There is no reason to expect this set of tools to be useful for *just any* purpose, in just any context, with just any constraints. We should perhaps expect computational explanation to be particularly successful for systems that have undergone a design process (including design by selection) to create a structure that efficiently generates appropriate outputs from inputs, because it is from disciplines creating and studying that kind of system that computational explanation draws most of its resources. But I won't pursue this here. The point is that although there is no reason to bar it a priori, many systems are unlikely to receive a successful or fruitful computational explanation.

4.3 Having abandoned *computation*, what makes an explanation *computational*?

As I've characterized it, only a special class of explanations count as computational ones. Where formalisms and conceptual resources drawn from the computing disciplines are used to meet explanatory needs like (A)–(C), you have a computational explanation. Otherwise you don't. But we might wonder whether this definition is sufficient. Consider, e.g., computer models of evolutionary processes that assume a sort of optimality to natural selection, and posit algorithms to achieve it. Say we have such a model, for which (B) and (C) are met to a large degree by conceptual resources from computing disciplines, but not to the same degree as in a typical computational model of (say) visual processing. In that case we should be happy to say the explanation is closer to a computational explanation, or more of a computational explanation, or is a more paradigmatic computational explanation. There is no reason to expect computational explanation to be a binary category. Since it is defined by the use of tools from the computing disciplines, those tools and those disciplines being fuzzily defined themselves, we should expect a fuzzy spectrum rather than strict criteria for counting as a computational explanation. This does not make it any harder to understand computational explanations or the source of their explanatory significance. That source was not their belonging to some strictly-defined category, it was their use of certain resources to meet particular needs. Some of those resources can be present — and therefore relevant to the explanations' force — while others are not, or are only to limited degrees.

I can now address an issue I postponed earlier. Some of the computational explanations I've mentioned, e.g. the ones to do with color vision, don't use formalisms drawn from computing disciplines. They use arithmetic. A certain set of retinal ganglion cells are described as computing S – (L + M), where the letters refer to the responses of different cone types. These explanations do, however, conceptualize the retina as following algorithms, one of the computational conceptual resources I pointed out above. And they may draw on knowledge from the computing disciplines, e.g., to do with the efficiency of different algorithms, the use of population-coding, data compression, and information-processing more generally (e.g., see Jameson et al. 2020, passim). To the extent that this terminology and its associated conceptual resources contribute to the our efforts to build process models of color vision, we have a case of computational explanation. There are more and less computational ways of describing color vision, and some may only weakly count as computational explanation. But there is no reason to demand that they count as full-blooded computational explanations when we have a good explanation of their function and success that doesn't require it.

I'll pause here to note a caveat. Though I raised these issues as questions of the definition of "computational explanation" I did not set out to explain *what computational explanation is*. I set out to explain *how and why computational explanation works*. To see how something works, you usually don't need to know what it is to be that thing. If you doubt this, try asking your mechanic for a rigorous definition of "engine," or a metaphysics of that category. So the above is intended to clarify the scope of my account and how it handles less paradigmatic cases — not to sharply define the category COMPUTATIONAL EXPLANATION. By way of illustration, consider Cisek's (1999) argument that cognition is non-computational because it consists largely of control processes that are not well-captured by classical computational thinking. On the Pragmatic Approach this does not raise a question of whether cognition is *really* computational or not, nor, to the present point, the a question of *whether the resources of control theory are really a part of computational explanation or not*. Instead, it raises the question of whether certain useful resources are neglected in cognitive science, what work they could be put to, and how to introduce them to do that work. There is just no need to define computational explanation so as to include *or* exclude explanations drawing on control theory. There is, instead, a need to investigate control theory and its potential usefulness in cognitive science, and to introduce it where it will be helpful. That project is ongoing (Richards and Lillicrap 2022), but has nothing to do with the metaphysics of computation or the definition of "computational explanation."

4.4 Metaphysical appendices

One final concern before I conclude. Consider a possible rejoinder to the Pragmatic Approach. You might try holding on to a metaphysics of computation while retaining the benefits of the Pragmatic Approach by giving the above account, and simply adding an appendix that says: whatever systems receive legitimate computational explanations according to the Pragmatic Approach *are thereby computers*. This lets us classify the brain as a computer without requiring our metaphysics to do any heavy lifting. Note that this approach — the Metaphysical Appendix Approach — accepts that the metaphysics of computation are irrelevant to understanding computational explanation. The proponent of this view just has some other reason to want a metaphysics of computation. (Maybe it's the kind of concept that *just can't fail* to correspond to a property, even if the use of the concept relies not at all on that property.) So they have accepted my main conclusion: if you want to understand how and why computational explanation works, you have no need for a metaphysics of computation. But then it's hard to think of a context in which the metaphysical appendix will play an important role, except perhaps if we're listing and describing all the properties our world contains — but recall that this list is, by hypothesis, irrelevant to science and the philosophy of science. Not that I have any problem with this variety of stamp collecting, but it has no role in answering the kind of questions I've taken up here.

And note: harmless though they may seem, appendices are liable to burst, and an approach that insists on a metaphysical appendix leaves itself open to complications. The resulting metaphysics of computation would be "stancey" and observer-relative. It is likely to be graded and fuzzy as well, given the discussion in section 4.3. These are not good objections, because on the Metaphysical Appendix Approach there is no reason to require a metaphysics of computation to be objective, observer-independent, etc. Perhaps a metaphysics *that is relied upon in science* should be all of those things, but one that exists only as an appendix for some other purpose cannot be held to these standards unless those other purposes demand it. But these objections, misguided as they are, bring with them a dialectical context — the context of the question, "what is it to be a computer," where they can be confused with good objections, or even grounds for rejection, and lead to the kind of debate we see between, e.g., Richards and Lillicrap (2022) and Brette (2022).

5 Conclusion

Let me summarize. Computational explanation in cognitive science works by using formalisms drawn from the computing disciplines, and the conceptual resources attendant on them, to construct process models that capture the causal structures in the brain that bring about cognitive capacities. It is successful as a general strategy because it serves the needs of cognitive science, and it is successful in specific instances, like any modeling practice or strategy, when it accurately describes the causal structures that bring about the behavior or capacity under investigation, and in a way that meets the various standards we have for scientific models and explanations. On the view I've described, there are limits on which computational explanations appropriately apply to which systems, preserving the explanatory significance of computation. Questions remain about the kinds of formalism computational thinking introduces, or should introduce, into cognitive science. But these questions should be answered along the lines of the Pragmatic Approach: not through the metaphysics of computation, but by a careful look at the resources that the explanations of interest involve and the goals they are intended to serve.

So far I've merely built up the Pragmatic Approach, letting the sense it makes of computational explanation stand as evidence for it. And I'm content to have simply gotten this approach on the table, ready for further discussion and refinement. It makes a sharp contrast to conventional approaches (e.g. Piccinini 2015; Shagrir 2022; Richards and Lillicrap 2022; Brette 2022) — a contrast that illuminates the assumptions of those approaches and (I've argued) their mistaken focus. But in the introduction I promised you a further argument against the Metaphysical Approach. That argument is this: if there is a working account that does without the metaphysics of computation, then to justify an account that accepts such a metaphysics, *you would need a reason to think that metaphysics is necessary*. The necessity of this metaphysics is rarely argued for, but the Metaphysical Approach posits and focuses on an entity — a kind, property, or category: COMPUTATION. An argument for this approach must specify some desiderata that the Pragmatic Approach doesn't achieve, but that the Metaphysical Approach does. Or it must give some other reason we should expect this property to exist, *and* to be relevant to cognitive science. Otherwise the Metaphysical Approach posits and spends time investigating the property for no apparent gain — it is explanatorily redundant and contributes nothing to our understanding of scientific explanation.

There is another way of coming at this point. To take the Metaphysical Approach is not only to posit a certain property, *computation*; it is to understand cognitive science and scientists as committed to the existence of that property, and to a specific understanding of that property. But these are commitments that cognitive scientists don't intend to, or appear to, make themselves.[20] Take just two examples of how mainstream cognitive scientists see their practice. Yamins and DiCarlo (2016) understand their preferred type of computational explanation as a way of "formalizing knowledge about the brain's anatomical and functional connectivity" so as to explain its cognitive capacities — not a metaphysical claim at all, and quite in line with the view I've defended here. And Richards et al. (2019) take deep learning and the computational explanations associated with it to involve the application of an explanatory/investigative "framework" involving specific types of models and hypotheses, principles about the causal structure of the brain, and strategies that the history of neural networks suggests for understanding complex systems. This is well-captured by the view I've defended, but to hold on to the Metaphysical Approach we would have to impose on this area of

[20]And cognitive scientists who do seem to make that commitment are quick to fall back to a pragmatic approach when issues like the triviality problem arise. A complication is that talk of *what computation is* may provide a useful framework for thinking about one's approach to computational models (thanks to Richard Lange and Rosa Cao for making this point in conversation). Thinking about what computation is could be just a way of thinking about computational models. I could hardly object to this if it really was just a methodological trick, rather than a substantive metaphysical commitment. But I take it that this framing almost always does involve metaphysical commitments. Regardless, if you can avoid those commitments, and see questions about the metaphysics of computation as a metaphysically non-committal shortcut to questions about computational explanation, we should have nothing to disagree about.

cognitive science commitments that it does not appear to make and that offer no obvious advantages over an approach that sticks more closely to scientific practice itself.

This is just one count on which the Pragmatic Approach is preferable to the Metaphysical Approach, but it reflects a broader range of considerations of significance to philosophers of cognitive science. To build the bridges between cognitive science and philosophy that most philosophers desire, it will be important to avoid, as far as possible, foisting the assumptions and definitions of one field onto the other. The Pragmatic Approach avoids at least one such foisting that the Metaphysical Approach does not. And, in fact, my version of the Pragmatic Approach does so by focusing on the context, practice, and function of computational explanation for cognitive scientists — another necessity for the bridge-building that philosophers have their hearts set on.

To conclude, computation provides a powerful, and crucial, lens on the brain. Philosophers of cognitive science, and many cognitive scientists themselves, have been duly impressed by the computational lens, but have failed to see it as a lens, instead understanding computation as a property of the brain itself. This is a natural enough mistake — a good lens is not perceived; it is perceived through. But if we forget we're looking through a lens, we will vastly misunderstand the things we see through it. For that matter, thinkers of a certain sort are liable to leave the lens on, turn to a chunk of rock, and shudder to discover that it has all the brain's computational properties too.[21] When, instead, we understand computation as a lens, we begin to see what it does to its target, what it occludes and makes salient, what it adds, what it blurs, what it brings into focus, and how, in turn, it makes the brain intelligible as the organ of the mind.

References

Bailer-Jones, D. M. (2002), 'Scientists' thoughts on scientific models', *Perspectives on Science* **10**(3), 275–301.

Barack, D. and Krakauer, J. W. (2021), 'Two Views on the Cognitive Brain', *Nature Reviews Neuroscience* **22**, 359–371.

Blohm, G., Kording, K. P. and Schrater, P. R. (2020), 'A how-to-model guide for neuroscience', *eNeuro* **7**(1), 1–12.

Brette, R. (2015), 'Philosophy of the spike: Rate-based vs. Spike-based theories of the brain', *Frontiers in Systems Neuroscience* **9**(November), 1–14.

Brette, R. (2022), 'Brains as Computers: Metaphor, Analogy, Theory or Fact?', *Frontiers in Ecology and Evolution* **10**(April), 1–5.

Cao, R. (2019), Computational Explanations and Neural Coding, *in* M. Sprevak and M. Columbo, eds, 'The Routledge Handbook of the Computational Mind', Routledge, pp. 283–296.

Carandini, M. (2012), 'From circuits to behavior: a bridge too far?', *Nature Neuroscience* **15**(4), 507–509.
URL: *http://dx.doi.org/10.1038/nn.3043*

Carandini, M. and Heeger, D. J. (2012), 'Normalization as a canonical neural computation', *Nature Reviews Neuroscience* **13**(1), 51–62.

[21] As in Chalmers (1996); Milkowski (2013); Piccinini (2015); Brette (2022), for just a few examples.

Chalmers, D. J. (1996), 'Does a Rock Implement Every Finite-State Automaton?', *Synthese* **108**, 309–333.

Chalmers, D. J. (2011), 'A Computational Foundation for the Study of Cognition', *Journal of Cognitive Science* **12**, 323–357.

Chirimuuta, M. (2014), 'Minimal models and canonical neural computations: the distinctness of computational explanation in neuroscience', *Synthese* **191**(2), 127–153.

Chirimuuta, M. (2019), Charting the Heraclitean Brain: Perspectivism and Simplification in Models of the Motor Cortex, *in* M. Massimi and C. D. McCoy, eds, 'Charting the Heraclitean Brain', Routledge, pp. 141–159.

Churchland, P. S. and Grush, R. (1999), Computation and the Brain, *in* R. Wilson and F. C. Keil, eds, 'The MIT Encyclopedia of the Cognitive Sciences', MIT Press, pp. 155–157.

Cisek, P. (1999), 'Beyond the Computer Metaphor: Behaviour as Interaction', *Journal of consciousness studies* **6**(11), 125–142.

Daniel, R., Schuck, N. W. and Niv, Y. (2015), 'How to divide and conquer the world, one step at a time', *Proceedings of the National Academy of Sciences of the United States of America* **112**(10), 2929–2930.

Danks, D. (2019), Probabilistic Models, *in* M. Sprevak and M. Colombo, eds, 'The Routledge Handbook of the Computational Mind', Routledge, pp. 149–158.

Devalois, R. and Devalois, K. (1993), 'A Multi-Stage Color Model', *Vision Research* **33**(8), 1053–1065.

Dewhurst, J. (2018), 'Individuation without Representation', *The British Journal for the Philosophy of Science* **69**, 103–116.

Dunbar, K. N. (2002), Understanding the role of cognition in science: the Science as Category framework, *in* P. Carruthers, S. Stich and M. Siegal, eds, 'The Cognitive Basis of Science', Cambridge University Press, pp. 154–171.

Edsel, A. (2016), *Breaking Failure*, FT Press, New Jersey.

Egan, F. (1999), 'In Defence of Narrow Mindedness', *Mind & Language* **14**(2), 177–194.

Egan, F. (2012), 'Metaphysics and Computational Cognitive Science: Let's Not Let the Tail Wag the Dog', *Journal of Cognitive Science* **13**(1), 39–49.

Egan, F. (2014), 'How to think about mental content', *Philosophical Studies* **170**, 115–135.

Egan, F. (2019), The nature and function of content in computational models, *in* M. Sprevak and M. Colombo, eds, 'The Routledge Handbook of the Computational Mind', Routledge, pp. 247–258.

Fodor, J. A. (1968), *Psychological Explanation: An Introduction to the Philosophy of Psychology*, Random House.

Fodor, J. A. (1975), *The Language of Thought*, Harvard University Press.

Frigg, R. and Hartmann, S. (2018), 'Models in Science'.

Frigg, R. and Nguyen, J. (2018), 'Scientific Representation'.

Gallistel, C. R. and King, A. P. (2009), *Memory and the Computational Brain*, Wiley-Blackwell.

Grothe, B., Pecka, M. and McAlpine, D. (2010), 'Mechanisms of sound localization in mammals', *Physiological Reviews* **90**(3), 983–1012.

Ham, Y.-G., Kim, J.-H. and Luo, J.-J. (2019), 'Deep learning for multi-year ENSO forecasts', *Nature* **573**(7775), 568–572.
URL: *http://dx.doi.org/10.1038/s41586-019-1559-7*

Hardcastle, V. G. (1996), *How to Build a Theory in Cognitive Science*, SUNY Press.

Hillis, D. W. (1998), *The Pattern on the Stone: The Simple Ideas that Make Computers Work*, Basic Books, New York.

Jacobs, G. H. (2014), 'The discovery of spectral opponency in visual systems and its impact on understanding the neurobiology of color vision', *Journal of the History of the Neurosciences* **23**(3), 287–314.

Jameson, K. A., Satalich, T. A., Joe, K. C., Bochko, V. A., Atilano, S. R. and Kenney, M. C. (2020), *Human Color Vision and Tetrachromacy*, Cambridge University Press.

Kriegeskorte, N. and Douglas, P. K. (2018), 'Cognitive computational neuroscience', *Nature Neuroscience* **21**(9), 1148–1160.
URL: *http://dx.doi.org/10.1038/s41593-018-0210-5*

Kucharski, A. (2016), 'Post-truth: Study epidemiology of fake news', *Nature* **540**(7634), 525.

Kwisthout, J. and van Rooij, I. (2020), 'Computational Resource Demands of a Predictive Bayesian Brain', *Computational Brain and Behavior* **3**(2), 174–188.

Lake, B. M., Ullman, T. D., Tenenbaum, J. B. and Gershman, S. J. (2017), 'Building machines that learn and think like people', *Behavioral and Brain Sciences* **40**.

Lazebnik, Y. (2004), 'Can a biologist fix a radio? Or, what I learned while studying apoptosis', *Biochemistry (Moscow)* **69**(12), 1403–1406.

Lycan, W. G. (1981), 'Form, Function, and Feel', *The Journal of Philosophy* **78**(1), 24–50.

Mancuso, K., Neitz, M., Hauswirth, W. W., Li, Q., Connor, T. B., Kuchenbecker, J. A., Mauck, M. C. and Neitz, J. (2010), 'Long-Term Results of Gene Therapy for Red-Green Color Blindness in Monkeys', *Invest. Ophthalmol. Vis. Sci.* **51**(13), 6292.

Marr, D. (1982), *Vision*, W.H. Freeman and Company.

Marr, D. (2010), *Vision*, MIT Press.

Matthews, R. J. and Dresner, E. (2017), 'Measurement and Computational Skepticism', *Nous* **51**(4), 832–854.

McClelland, J. L., Rumelhart, D. E. and Hinton, G. E. (1986), The Appeal of Parallel Distributed Processing, *in* D. E. Rumelhart, J. L. McClelland and P. R. G. The, eds, 'Parallel Distributed Processing', MIT Press, pp. 3–44.

McCulloch, W. S. and Pitts, W. (1943), 'A logical calculus of the ideas immanent in nervous activity', *The Bulletin of Mathematical Biophysics* **5**(4), 115–133.

Milkowski, M. (2013), *Explaining the Computational Mind*, MIT Press.

Miller, M. (2014), *Minds Online: Teaching Effectively with Technology*, Harvard University Press, Cambridge MA.

Neander, K. (2017), *A Mark of the Mental*, MIT Press.

Nersessian, N. J. (2002), The cognitive basis of model-based reasoning in science, *in* P. Carruthers, S. Stich and M. Siegal, eds, 'The Cognitive Basis of Science', Cambridge University Press, pp. 133–153.

Neumann, J. v. (1958), *The Computer and the Brain*, Yale University Press.

Nityananda, V. and Read, J. C. (2017), 'Stereopsis in animals: Evolution, function and mechanisms', *Journal of Experimental Biology* **220**(14), 2502–2512.

Peacocke, C. (1994), 'Content, Computation and Externalism', *Mind & Language* **9**(3), 303–335.

Piccinini, G. (2015), *Physical Computation: A Mechanistic Account*, Oxford University Press.

Putnam, H. (1991), *Representation and Reality*, MIT Press.

Pylyshyn, Z. W. (1984), *Computation and Cognition*, MIT Press.

Pylyshyn, Z. W. (1993), Computing in Cognitive Science, *in* M. I. Posner, ed., 'Foundations of Cognitive Science', MIT Press, pp. 49–92.

Richards, B. A. and Lillicrap, T. P. (2022), 'The Brain-Computer Metaphor Debate Is Useless: A Matter of Semantics', *Frontiers in Computer Science* **4**(February), 1–8.

Richards, B. A., Lillicrap, T. P., Beaudoin, P., Bengio, Y., Bogacz, R., Christensen, A., Clopath, C., Costa, R. P., de Berker, A., Ganguli, S., Gillon, C. J., Hafner, D., Kepecs, A., Kriegeskorte, N., Latham, P., Lindsay, G. W., Miller, K. D., Naud, R., Pack, C. C., Poirazi, P., Roelfsema, P., Sacramento, J., Saxe, A., Scellier, B., Schapiro, A. C., Senn, W., Wayne, G., Yamins, D., Zenke, F., Zylberberg, J., Therien, D. and Kording, K. P. (2019), 'A deep learning framework for neuroscience', *Nature Neuroscience* **22**(11), 1761–1770.

Richmond, A. (n.d.*a*), 'Computational Externalism', *Forthcoming* .

Richmond, A. (n.d.*b*), 'What is a Theory of Neural Representation For?', *Forthcoming* .

Samuels, R. (2019), Classical Computational Models, *in* M. Sprevak and M. Colombo, eds, 'The Routledge Handbook of the Computational Mind', Routledge, pp. 103–119.

Sánchez, V. G. (n.d.), 'What Bayesian Angels have to do with Human Cognition', *Forthcoming* .

Scheutz, M. (2012), 'What it is not to Implement a Computation: A Critical Analysis of Chalmers' Notion of Implementation', *Journal of Cognitive Science* **13**(1), 75–106.

Searle, J. R. (1992), *The Rediscovery of Mind*, MIT Press.

Sejnowski, T. J., Koch, C. and Churchland, P. S. (1988), 'Computational Neuroscience', *Science* **241**(4871), 1299–1306.

Shadmehr, R. and Wise, S. (2005), *The Computational Neurobiology of Reaching and Pointing*, MIT Press.

Shagrir, O. (2001), 'Content, Computation and Externalism', *Mind* **110**(438), 369–400.

Shagrir, O. (2018), 'In defense of the semantic view of computation', *Synthese* (January).
URL: *https://doi.org/10.1007/s11229-018-01921-z*

Shagrir, O. (2022), *The Nature of Physical Computation*, Oxford University Press, New York.

Shea, N. (2013), 'Naturalising Representational Content', *Philosophy Com* **8**(5), 496–509.

Simon, H. A. and Newell, A. (1973), ‘Human Problem Solving: The State of the Theory in 1970’, *American Psychologist* **26**(2), 145–159.

Smith, B. C. (1999), Computation, *in* R. A. Wilson and F. C. Keil, eds, ‘The MIT Encyclopedia of the Cognitive Sciences’, MIT Press, pp. 153–155.

Sower, V. E., Duffy, J. A. and Kohers, G. (2008), ‘Ferrari’s Formula One Handovers and Handovers From Surgery to Intensive Care’, *The American Society for Quality* (August), 1–5.
URL: *www.asq.org*

Sprevak, M. (2019), Triviality arguments about computational implementation, *in* ‘The Routledge Handbook of the Computational Mind’, Routledge, pp. 175–191.

Sun, D., Lv, J. and Waller, S. T. (2011), ‘In-depth analysis of traffic congestion using computational fluid dynamics (CFD) modeling method’, *Journal of Modern Transportation* **19**(1), 58–67.

Sun, R. (2008), Introduction to Computational Cognitive Modeling, *in* R. Sun, ed., ‘The Cambridge Handbook of Computational Psychology’, Cambridge University Press, pp. 3–20.

Thoreson, W. B. and Dacey, D. M. (2019), ‘Diverse Cell Types, Circuits, and Mechanisms For Color Vision in The Vertebrate Retina’, *Physiol Rev* **99**, 1527–1573.

Waismann, F. (1968), Verifiability, *in* R. Harré, ed., ‘How I see Philosophy’, Palgrave Macmillan, chapter 2.

Yamins, D. L. and DiCarlo, J. J. (2016), ‘Using goal-driven deep learning models to understand sensory cortex’, *Nature Neuroscience* **19**(3), 356–365.