How Computation Explains

Andrew Richmond

**Abstract.** This paper discusses computational cognitive science: the use of tools, concepts, and strategies from the computer sciences to investigate the brain. Philosophers have assumed that computational cognitive science, and the *computational explanations* it provides, claim that the brain *computes*, in a sense to be specified by the metaphysics of computation. That metaphysics, by revealing *what it is* to compute, and, therefore, what we attribute to the brain when we say it computes, is supposed to show how and why computational explanation works. In contrast, I defend a pragmatic approach: I show how and why computational explanation works by describing the resources the notion of computation brings with it, and how those resources serve cognitive science. Specifically, I argue that the notion of computation introduces concepts and formalisms that complement cognitive science’s modeling goals. This allows us to understand computational explanation without ever broaching the metaphysics of computation, or debating *what it is* to compute.

# **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]](#footnote-1)

Perhaps this has become commonplace enough to dull our critical instincts, but if we dwell for a moment on the explosion of computational explanations over the past half-century, some questions arise. What arethese explanations? How do they work? What distinguishes them from other kinds of explanation? And why have they seemed to be 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?

The received view is that computational explanations work because the brain *is* a computer. The brain supports depth perception, e.g., by literally computingretinal 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. The Metaphysical Approach is characterized by an assumption Chalmers makes explicit: “[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?*” (2011, 325; *cf.* Piccinini 2015; Shagrir 2022).(It’s no different whether one thinks that computational explanations work because the brain is *a computer* or just because it *computes*. The questions that arise are perfectly analogous: *What is ~~a computer~~ computing? What features make something ~~a computer~~ compute?* So I won’t distinguish between the two views here.)

Note that the Metaphysical Approach doesn’t want a theory of computation like Turing’s. That’s a theory of computable functions, i.e., functions for which there exist effective methods or algorithms, and the nature of those algorithms. Those functions and algorithms are formal, abstract things. 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 compute addition? 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 that would make the brain a computer, and that would 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 article 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 thinking of the brain as a computer *is* metaphorical, it isn’t *just* a metaphor — it’s one of the most widespread and apparently successful explanatory approaches in recent science. That needs explaining, and “it’s just a metaphor” isn’t an explanation.[[2]](#footnote-2) 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 aiming to show how and why computational explanation works. 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 workings and apparent success of computational explanation in some other way. This paper is an attempt to discharge that burden, with resources from less controversial domains than the metaphysics of computation: I’ll appeal only to the kind of things computational *notions* allow us to do as we investigate, and especially model, the brain.

I’ll further 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 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. 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: it still aims to understand computational explanation, but it does so without saying *what computation is*, asking instead what resources the concept of computation introduces into cognitive science, and how those resources serve cognitive science’s goals. I’ll develop the Pragmatic Approach in response to some objections in section 4, before concluding in section 5.

# **2. Triviality**

In the examples I began with, the brain is not merely modeled computationally, like the weather is. Weather models predict the future behavior of the weather, but often not the internal processes that bring it about (e.g., Ham et al. 2019). 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]](#footnote-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, and the main hurdle for philosophers answering that question is the *triviality problem*. Many seemingly plausible accounts of what it is to compute end up counting too many systems as computers, and counting any given system as computing too many things. This section will describe that problem in more detail to bring out the distinctive features of theMetaphysical Approach — features the Pragmatic Approach will not share.

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. 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*.

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]](#footnote-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 the problem is not limited to three-step computations.[[5]](#footnote-5) 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.

First, much debate in cognitive science is *between* computational models.E.g., it is because the brain performs cone-opponent computations *rather than* simple cone summations that it supports color vision (Jacobs 2014). If it trivially performed both computations, neither model could make a unique prediction, because each would have to allow that the brain also performed the other model’s computations. 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 a brain as with a rock) than the history of cognitive science would suggest. And second, even if the brain performed a limited set of computations, it would be a problem if too many other systems performed them too. If any old rock implements an addition algorithm, then the brain’s implementing that same algorithm can’t explain why it, and not any rock, supports mathematical cognition. If we allow the brain to compute too many things, or if we allow too many things to compute, we sever the tie between computations and the capacities they’re supposed to explain: we underminecomputational explanation, and make its success mysterious.

If we’re approaching computational explanation through the metaphysics of computation, this problem must be solved by a narrower definition of computation: one that limits which systems implement which computations, in a way that preserves the scientific role of the notion of computation. Whether we ground computation in causal properties (Chalmers 1996), teleological ones (Milkowski 2013; Piccinini 2015), or representational ones (Peacocke 1994; Shagrir 2018), the problem the Metaphysical Approach revolves around is the same: defining physical computation so as to include all the right systems and exclude all the wrong ones. This is supposed to be the first step in explaining how computational explanation works, because it tells us what exactly computational explanations attribute to the brain.

What is distinctive about the Pragmatic Approach is that it will make the triviality problem, and the associated puzzles about *what it is* to compute, irrelevant. The main desideratum for the Pragmatic Approach is the same as for the Metaphysical Approach: to explain how computational explanation works, and why it seems to work so well. But the Pragmatic Approach does not achieve this through a metaphysics of computation. Instead, it looks to the scientific goals computational explanation serves and how it serves them, focusing especially on the resources that computational notions introduce to cognitive science. If this suffices to explain computational explanation, it will have shown that no metaphysics of computation is necessary. That’s what I’ll try to show in the next section.

First, though, let me make a digression that I’d rather not relegate to a footnote. Egan is widely interpreted as an advocate of the simple mapping account as I’ve described it, but this is not the only possible interpretation of her work. It is also plausible that, for Egan, mappings do not constitute a metaphysics of computation, but rather a necessary condition on *the success of computational explanations*. In fact, this is how Egan herself describes her view (Egan, in correspondence). If that’s right, then, while the metaphysical version of the simple mapping account is a common starting-point and foil for other accounts of computation, it isn’t Egan’s view. Egan’s view would be compatible with my own, and this paper could be seen as building on her pioneering work.

# **3. Domain transfer and the Pragmatic Approach**

The goal of this section is to build an account of computational explanation that explains how and why it works, but without pulling us into debates about the metaphysics of computation, or about whether the brain *really is* a computer.

*3.1 Making room for the pragmatic approach*

In the previous section, I distinguished between merely predictive computational models and genuine computational explanations. 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*.[[6]](#footnote-6) (Simon and Newell 1973, 147, my italics)

But process models do not necessarily 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 some *inherent property of calculus-implementation* in 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; Panchal & Jack, 2022). These models don’t require or assume that traffic literally is a fluid, or that disinformation is, by some criteria, a virus.

The fluid and virus examples are cases 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). The doctors are not performing a ballet, and their patients are not assumed to belong to the category Ferrari or McLaren. We don’t 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 what goals this allows them to serve in another domain.

To summarize: the negative point of this paper, which I’ve been stressing so far, is that it’s premature to say 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 how computational explanation works, and how it serves cognitive science. If computational explanation works like a typical domain transfer, the project Chalmers describes is unnecessary. The positive point of this paper, to which I’ll turn now, is 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: capacities to detect and distinguish between stimuli, to decide on a course of action, to navigate spatial and social environments, and so on. These are capacities to produce appropriate behavior in a range of environments and given a range of inputs. And cognitive science, at least since the rejection of behaviorism, aims to explain these capacities by appealing 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.

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 explainbehavior using that description. This means we need, at least:

**(LF)** A Language or Formalism with which to describe the causal structures in the brain that support cognitive capacities.

**(CR)** Conceptual Resources with which to form questions, hypotheses, and explanations regarding those structures and the way they support cognitive capacities.

The need for expressive languages and formalisms is widely discussed. See Lazebnik (2004), e.g., on the importance of a good formalism in biology for making predictions, framing hypotheses, revealing 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: an appropriate conceptual framework is required to ensure that our formalism is appropriate to our subject matter: together, the formalism and our conceptual resources should facilitate *theoretically useful descriptions* of the causal structures we seek (Lazebnik 2004). That is to say, cognitive science does not just need a language to describethe brain, but also to state its explananda, and to frame relevant questions and hypotheses about those explananda. A formalism borrowed from particle physics might describe the brain well in many respects, but it would likely be difficult or impossible, within that formalism, to state explananda having to do with an organism’s capacity to memorize strings of words, or to frame hypotheses about how the brain’s causal structure supports deductive reasoning.

So, satisfying (LF) and (CR) would give us a set of tools for describing the causal structures in the brain that bring about its behavior and support its cognitive capacities, and for posing relevant questions and hypotheses about how they do so.[[7]](#footnote-7) (It’s also crucial that our formalisms and conceptual resources allow us to *test* our hypotheses. But this is essentially the problem of giving them empirical content, which I’ll treat in section 4.) My claim is that computational notions introduce into cognitive science a set of formalisms and conceptual resources satisfying (LF) and (CR), and as such they contribute to the goals of cognitive science. As long as they do this well, they facilitate good explanations. And since those explanations would be descriptions of causal structures — not subsumptions 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 previous examples. The task, then, is to see whether and how (LF) and (CR) are satisfied by computational formalisms and concepts.

Let’s start by getting clearer about the formalisms and concepts computational explanations use. An immediate difficulty is that it’s unclear how formalisms and concepts should be categorized. Which formalisms count as *computational* is not straightforward, and seems to depend on how tools from computer science are exported into new domains (Smith 1999; Kriegeskorte and Douglas 2018). So I’ll lean on the way the notion of computation is used in cognitive science: I’ll allow that whatever, for cognitive scientists, is generally counted as a computational explanation, *is* a computational explanation. This will include explanations that use the formalism of Turing machines, the formalisms of finite and combinatorial state automata, the formalisms that describe perceptrons and artificial neural networks, as well more generic forms of description like wiring diagrams. What these formalisms have in common is that they tend to invoke the devices built by computer engineers (e.g., wiring diagrams), the programs designed by computer programmers (e.g., search algorithms), or the mathematical structures investigated in computer science (e.g., Turing machines or Hopfield networks). 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, which I’ll call *computational formalisms*, are computational explanation’s solution to (LF). I’ll take the same approach to conceptual resources: the conceptual resources that are attendant on computational formalisms, or that are otherwise drawn from the computing disciplines, I will call *computational concepts*: these are computational explanation’s solution to (CR). To understand how and why computational explanation works, then, we need to understand how it uses these formalisms and concepts, and what makes them suited to achieving (LF) and (CR).

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. 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 which circumstances will put the wire in a 1-state or a 0-state), we can exploit that distinction to build more complex functions like logic gates.

There is no need to belabor the utility of functional abstraction for engineering; what’s important is that it helps to reverse-engineer 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 — 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, 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 it remains 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 compose those functions (Kriegeskorte and Douglas 2018). And to do this we need a way to abstract 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, and these are many of the same questions we have about the brain (Kriegeskorte and Douglas 2018, Box 3). So computational formalisms give us a toolkit for functional abstraction that is particularly well-suited to our questions about the brain.

Computational formalisms and the conceptual frameworks they bring with them also lend themselves to two paradigmatic types of description: 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 processes between input and output. 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 cell, 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 a type 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 determining its low-level features (lines, shapes), using those to determine its intermediate and eventually high-level features like object types (e.g., *dog* or *cat*). This was also a goal of classical computational modeling (Marr 1982). In computational neuroscience, descriptions like these are essential to understanding how the brain proceeds from sensory stimulation to sophisticated high-level categorization and behavior based on it, and conceptual frameworks drawn from neural networks (and other sources in the computing disciplines, Marr 1982) have illuminated the relevant brain processes by providing useful and increasingly accurate models of these processes (Richards et al. 2019).

Another kind of hierarchical description is *compositional*, rather than algorithmic. A compositional hierarchy does not necessarily derive higher- and higher-level features; it consists of a small set of simple functions that compose more and more complex ones. E.g., the processes involved in different cognitive capacities may rely on a shared “set of standard (canonical) neural computations: combined and repeated across brain regions and modalities to apply similar operations to different problems” (Carandini 2012). This kind of hierarchical description is useful for cognitive science primarily because our understanding of simple neural functions guides anatomical investigation into basic units and circuits, making relevant aspects of their causal structure salient (Carandini and Heeger 2012), and because it guides model-building, since our goal is often to model how high-level capacities results from low-level structures (Yamins and DiCarlo 2016). Computational formalisms, and the concepts associated with them, allow us to give these descriptions of the brain’s causal structure, and are particularly well-suited to that task: 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); and they are developed to create and make intelligible complex relationships between different levels of abstraction.

Let me pause here to make a point that will recur over the following pages. It cannot be taken as *given* that cognitive science should be explaining behavior by appealing to the internal processes that generate it, or by decomposing capacities into a set of lower-level functions assigned to particular components in the brain. Those goals are what make algorithmic and hierarchical descriptions — and therefore the computational formalisms and concepts that facilitate them — so useful. But those goals are disputed, e.g., by Chemero (2011) and Anderson (2014), respectively. My intention is not to argue *for* computationalism, i.e., to argue that computation gives us the correct way to think of the brain. That would call for a very different sort of paper. My intention is to show how computational explanation works, and why it supports cognitive science’s current goals, given its current assumptions and context. I am not arguing that we really should be trying to accomplish those goals, or that those assumptions are correct. E.g., recall that I didn’t reject behaviorism at the start of section 3.2, but only noted that computational explanation is predicated on its rejection. We can productively debate computationalism only once we understand how it works and what it’s doing: it’s *those* questions I’m answering.

Moving on: 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 solution to (LF). Of course, I’ve also been talking about (CR) this whole time. E.g., I didn’t just discuss how the computational formalisms allow us to describe hierarchies and algorithms: I also pointed out that *conceptualizing* the brain in algorithmic and hierarchical terms guides investigation into the brain, and allows us to answer various questions about the its causal structure. But the conceptual resources allowing computational explanation to answer (CR) do not just tag along with computational formalisms: they can also be pulled independently from the computing disciplines. This is not surprising: the assimilation of one system under the conceptual scheme developed for another is a widespread and natural part of science (Dunbar 2002; Nersessian 2002). And, though my examples here will be brief, it is easy to see that computational concepts serve concrete goals in cognitive science. Considerations of *algorithmic complexity* drive discussions about the appropriateness of Bayesian models of the brain, identifying the kind of complexity that would make a computational causal model appropriate or inappropriate (Kwisthout and van Rooij 2020). Similarly, considerations of *computational efficiency* drove early debates in cognitive neuroscience (McClelland et al. 1986), and considerations of *computational cost* drive current discussions of navigation and route planning, providing fruitful and easily-formalized ways to frame questions about the resources an organism can devote to its problems (Daniel et al. 2015). It is because we think of the brain in computational terms that we investigate its *canonical computations*, i.e., its basic low-level operations, with all of the benefits noted above. It is because we understand the properties of *recurrent connections in neural networks* that we look for recurrent connections among neurons, understanding that models incorporating recurrent connections, if they are accurate, can explain certain capacities better than other models (Richards et al. 2019).

If this seems to belabor the obvious, recall that the point isn’t that it’s useful to think of the brain as a computer. The point is to say what we do when we think of the brain as a computer: we apply formalisms and conceptual schemes to a system, because they provide concrete scientific benefits, especially for modeling. And if that’s right, we can make sense of computational explanation without worrying about whether brain *is* a computer, and without entering into vexed questions about what that means. Computational explanation works by introducing formalisms and concepts from the computing disciplines, and in doing so it supports cognitive science by providing, among other things: explanatorily relevant functional abstractions; descriptions in the fruitful terms of algorithms and hierarchies; tight connections between our descriptions of the brain’s causal structure and cognitive science’s questionsabout it; and relevant and fruitful ways to frame those questions, along with their answers. In turn, this provides explanations of how the brain supports cognition that are natural, powerful, and especially sensitive to our explanatory goals.[[8]](#footnote-8)

Pulling back from the details, the point is that computational concepts and formalisms help us construct process models that capture the causal structures in the brain that bring about its cognitive capacities. If this is what they do, then understanding how they do it or why computational explanation is successful does not require us to ask whether the brain *is a computer*, or give a theory about what that means. Computational explanation can serve cognitive science like virological explanations serve the study of disinformation, and like many other formalisms and conceptual frameworks serve science: by providing resources that answer to science’s goals and needs.

**4. Objections**

The bulk of the Pragmatic Approach is on the table. I’ll now consider some objections.

*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 I don’t invoke. 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 says about its target system. That is, I haven’t yet placed any constraints on the *empirical content* of computational explanations. So, 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 the model is correct as long as the rock runs through time-slices that have *just some mapping* to the stages of the model’s algorithm?

But this is an implausible development of the triviality problem. Two constraints ensure that what computational explanations say about the causal structure of their target systems is non-arbitrary. First, note that since I’m treating computational explanation as a modeling practice, this is just a special case of a more general problem: in virtue of what does any model say what it says about its target system (Frigg and Nguyen 2018; Frigg and Hartmann 2018)? I don’t propose to answer this question here, but we can be sure that it is not arbitrary what scientific models in general say about their target systems, and there’s no reason that the models generated by computational explanations would be idiosyncratic in this respect.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. And note that the problem of model reference is not generally solved by criteria for a target system’s membership in a particular category, like *calculizer*, *virus*, and so on. There is no reason to think the models generated by computational explanations would be idiosyncratic in this respect either.

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 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 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, along with our explanatory goals. To explain a memory task, our neuroscientist 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 whether to model population activity or individual neurons and their connections (Barack and Krakauer 2021). Just as we should expect, the empirical content of computational models is constrained by the same kind of empirical considerations that constrain the empirical content of any other models.

So: why can’t I give a computational process model of a rock as performing addition, and say the model is correct as long as the rock runs through time-slices that have *just some mapping* to the stages of the model’s algorithm? For the same unmysterious reasons that I can’t give a process model of the solar system using calculus, and say that the model is correct as long as the solar system runs through time slices that have *just some mapping* to the model. That’s not how scientific models work, and this revival of the triviality problem is either an implausible skepticism about model reference in general,[[9]](#footnote-9) or an unsupported conviction that scientists using *computational* models have some unique difficulty understanding what exactly they’re saying about the brain.

*4.2 Applying computational explanation*

There are circumstances where computational explanations are appropriate, and ones where they aren’t. Without the category *computer*, can we say which systems should receive computational explanations and which shouldn’t? 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) the weather, because accommodating this does not require a revisionary metaphysics according to which weather patterns compute. We should note, however, that computational explanations are in fact not helpful in most cases, at least 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 anypurpose, in just any context, with just any constraints.

More constructively: we should expect computational explanation to be particularly successful for systems that have undergone a design process (including 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. More broadly, though, we should expect computational explanation to be successful when the form of our questions makes computational explanation’s resources (hierarchy, algorithm, etc.) germane, and where the explanatorily relevant features of the target system’s causal structure are features that computational resources model well.

As a simplistic example, imagine we decide that the right questions in cognitive science are not about the brain’s internal causal structure, but about its couplings with its environment (Chemero 2011). This would make computational explanations less promising, because it would make the resources that computational explanation introduces less germane to our questions.[[10]](#footnote-10) Or imagine it turns out that, to understand cognitive capacities, it is most useful to model individual neurons at a fine grain, and especially to model saltatory action potentials. That would make certain (old-fashioned) computational explanations more appropriate, because it would make certain (old-fashioned) computational formalisms and concepts especially useful in modeling (as in McCulloch and Pitts 1943; Neumann 1958).[[11]](#footnote-11) What matters is whether computational resources are a good fit for our goals, and that depends on both the nature of those goals and the details of the brain. These kinds of consideration are perfectly capable of ruling computational explanation in or out, without ever implicating questions about whether the brain *is* a computer.

One other factor that might make computational explanations appropriate for cognitive science would be a computational/functional understanding of *mind* or *cognition*.[[12]](#footnote-12) Insofar as our understanding of the *mind* dictates the questions that science must ask of the *brain*, then, if we are committed to framing our questions about the mind in computational terms, it would be natural for science to frame and answer questions about the brain in computational terms. But note that this doesn’t impose any metaphysical commitments on the relevant scientific work, or raise questions about whether the brain is or isn’t a computer. This is just a more obvious example of how computational explanations can be a good fit for certain questions and not others. I mentioned above that questions about *structure*, as opposed to *couplings*, might be best answered in computational terms. Likewise, and for more obvious reasons, questions *framed in computational terms* might be best answered in computational terms.[[13]](#footnote-13)

As I’ve stressed, the Pragmatic Approach explains *how* computational explanation works, not *whether* it works. But, by highlighting the features that would make computational explanations successful, it does help direct the debate about whether computational explanations work, or whether they are appropriate for cognitive science: it asks us to handle that debate by considering which questions we should be asking; which aspects of the brain are most responsible for the phenomena we’re studying; and which kinds of model most meaningfully connect the details of the brain to the capacities we’re explaining and the questions we’re asking. These are, I submit, more fruitful directions than the winding paths we tend to travel when we ask whether the brain *really is* a computer (e.g. Searle 1990; Richards and Lillicrap 2022; Brette 2022).[[14]](#footnote-14)

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

As I’ve characterized things, only a certain class of explanations count as computational ones. Where formalisms and concepts from the computing disciplines are used to meet explanatory needs like (LF) and (CR), you have a computational explanation. Otherwise, you don’t. But is this definition sufficient? Consider, e.g., a computer model of evolution that assumes a sort of optimality to natural selection, and posits algorithms to achieve it. This model would draw fewer resources from computing disciplines than a typical computational model of visual processing. It is a computational explanation or not?

We should be happy to say the explanation is *more* of a computational explanation, or is a more *paradigmatic* computational explanation. There is no reason to expect computational explanation to be a simple binary category. Since it is defined by the use of tools from the computing disciplines, those tools and those disciplines not being strictly defined themselves (Smith 1999; Rorty 1979, 331; Kuhn 1977, xvi), we should expect a fuzzy spectrum rather than strict criteria for counting as a computational explanation. Most importantly, this does not make it any harder to understand computational explanations or the source of their explanatory significance, because that source was never 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.

A good example is the explanation of color vision, which doesn’t tend to use formalisms drawn from computing disciplines: it uses arithmetic. A certain set of retinal ganglion cells are described as computing, e.g., *aL(λ) – bM(λ)* (Shevell and Martin 2017).[[15]](#footnote-15) These explanations do, however, conceptualize the retina as following algorithms, one of the computational conceptual resources I pointed out above. And they draw on other knowledge from the computing disciplines, e.g., to do with the efficiency of different algorithms, methods for data encoding and compression, and so on (Jameson et al. 2020). To the extent that these resources contribute to our explanation of color vision, we have a stronger case of computational explanation. To the extent that they don’t, we have a weaker case. There will be plenty of examples like this, and we can understand their function and success just as we do with any explanatory process model: by seeing how they use their various resources (some computational, some not) to describe the causal structure of their target system.

*4.4 Metaphysical appendices*

One final concern. 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 successful computational explanations according to the Pragmatic Approach *are thereby computers*. This lets us classify some systems, perhaps including the brain, as computers — but without the metaphysics doing any heavy lifting.

Note that this 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 metaphysical category, even if the role of the concept has nothing to do with that category?) We may be able to think up contexts where the metaphysical appendix willbe important, even if it’s just in listing and describing the properties our world might contain. It’s not that I have any problem with this kind of stamp collecting, but it has no role in answering the kind of questions I’ve taken up here. As long as we agree that the metaphysics is irrelevant for the purpose of understanding cognitive scientific explanation, we agree on what matters.

And note: harmless though they may seem, appendices are liable to burst, and a strategy that insists on a metaphysical appendix leaves itself open to complications. The resulting metaphysics of computation would say, among other things, that which systems are computers depends on our explanatory goals and current state of knowledge (since whether a system receives a successful computational explanation depends on those things). This will be anathema to the typical metaphysician of computation, and is liable to be seen as a death-blow for the account (e.g., Piccinini, 2015, Chapter 1). I’ve given you the account post-appendectomy. Re-framing it as a deflationary metaphysics of computation will only distract from the real purpose of the account — to explain computational explanation— and set it up to be dismissed for failing to meet an unrelated set of metaphysical desiderata.[[16]](#footnote-16)

**5. Conclusion**

Computational explanation in cognitive science works by using formalisms and conceptual resources drawn from the computing disciplines to construct process models that capture the causal structures in the brain that bring about cognitive capacities. It is so widespread and apparently successful as a general strategy because it serves cognitive science well, given cognitive science’s particular needs and goals. And it is successful in any particular case so long as it accurately describes the causal structures that bring about the behavior or capacity under investigation in a way that meets the goals and standards cognitive science sets for its explanations. And on the Pragmatic Approach there are limits on how and when computational explanations can be successfully applied, and on which computational explanations appropriately apply to which systems, preserving the explanatory significance of computation.

This approach makes sense of computational explanation while avoiding fraught debates over what it is to be a computer, the idiosyncratic treatment of computational explanations as involving metaphysical commitments that similar forms of explanation don’t, and the digressions from scientific practice that result from a focus on those debates and those commitments. It also hews more closely to neuroscience’s self-conception: computational cognitive scientists don’t intend to, or appear to, make the commitments that the Metaphysical Approach thinks computational explanation involves. 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 and investigative “framework” using specific kinds 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 Pragmatic Approach, and not by the Metaphysical Approach, which would impose on this area of 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.

This is just one count on which the Pragmatic Approach is preferable to the Metaphysical Approach, but it reflects something especially important for 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 are working towards.

To conclude, computation provides a powerful, and perhaps 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.[[17]](#footnote-17) 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**

Anderson, M. L. (2014), *After Phrenology*, MIT Press.

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.

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

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.

Chemero, A. (2011), *Radical Embodied Cognitive Science*, MIT Press.

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.

Curtis-Trudel, A., & Symons, J. (n.d.). What is a computer? Philosophy of computation after philosophy of mind. *Ms.*

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.

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. (2014), ‘How to think about mental content’, *Philosophical Studies* 170, 115–135.

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.

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.

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

Kuhn, T. S. (1977), *The Essential Tension: selected studies in scientific tradition and change*, University of Chicago Press.

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., *et al*. (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., *et al*. (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.

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.

Mollo, D. C. (2021), ‘Against Computational Perspectivalism’, *British Journal for the Philosophy of Science* 72(4), 1129–1153.

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.

Panchal, R., & Jack, A. (2022). The contagiousness of memes: containing the spread of COVID-19 conspiracy theories in a forensic psychiatric hospital. *BJPsych Bulletin*, *46*(1), 36–42. https://doi.org/10.1192/bjb.2020.120

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.

Rescorla, M. (2017). The Computational Theory of Mind. In *Stanford encyclopedia of philosophy*. https://doi.org/10.1145/544317.544327

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., *et al.* (2019), ‘A deep learning framework for neuroscience’, *Nature Neuroscience* 22(11), 1761–1770.

Rorty, R. (1979), *Philosophy and the Mirror of Nature*, Princeton University Press.

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

Samuels, R. (2019). Classical Computational Models. In M. Sprevak & M. Colombo (Eds.), *The Routledge Handbook of the Computational Mind* (pp. 103–119). Routledge.

Sánchez, V. G. (n.d.), ‘What Bayesian Angels have to do with Human Cognition.’ *Ms*.

Searle, J. R. (1990), ‘Is the Brain a Digital Computer?’, *Proceedings and Addresses of the American Philosophical Association* 64(3), 21–37.

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

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

Shagrir, O. (2018), ‘In defense of the semantic view of computation’, *Synthese* (January).

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

Shevell, S. K., and Martin, P. R. (2017), ‘Color Opponency: Tutorial,’ Journal of the Optical Society of America, A, vol. 34, no. 7, 1099–108.

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*

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.

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.

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

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). [↑](#footnote-ref-1)
2. Even you reject the computational approach, its *apparent* success calls for explanation. [↑](#footnote-ref-2)
3. This is not the only use for computational models. 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. [↑](#footnote-ref-3)
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. [↑](#footnote-ref-4)
5. If we carve up the rock’s states more finely, we can map it to computational structures like whole Turing machines or neural networks. [↑](#footnote-ref-5)
6. Marr (1982, 23) expresses a similar sentiment, as do Fodor (1968), Kriegeskorte and Douglas (2018), and Nancy Dice (quoted in Bailer-Jones 2002). [↑](#footnote-ref-6)
7. This is not all that formalisms and conceptual resources should do for a field of science. But these two desiderata will reveal much of the way computational explanations function and the reasons they succeed. [↑](#footnote-ref-7)
8. It is tempting to point out that some computational resources (like early neural networks) were inspired by the brain. Domain transfer may be complex and involve significant back-and-forth of that sort. My point has been that the back-and-forth involves the sharing and development of formalisms and concepts that serve concrete explanatory purposes, rather than metaphysical commitments. [↑](#footnote-ref-8)
9. Matthews and Dresner (2017) similarly argue that triviality arguments about computation reduce to triviality arguments about *any attribution of numerical properties to a physical system*. The advantage of the Pragmatic Approach is that it shows exactly what’s wrong with triviality arguments: those arguments only get off the ground if computational explanations are *metaphysical commitments*, rather than more typical models. [↑](#footnote-ref-9)
10. Not, of course, because it implies that the brain isn’t a computer. It hasn’t said anything *about* the brain, just that one set of questions would serve us better than another. [↑](#footnote-ref-10)
11. Not, of course, because it would inform us that the brain is a computer. We already know about the saltatory action potential; the question is not about whether the brain has that feature (and therefore counts as a computer), but whether we should be trying to model that feature. [↑](#footnote-ref-11)
12. Thanks to an anonymous reviewer for raising this point. [↑](#footnote-ref-12)
13. Things are different if the “computational/functional understanding of *mind* or *cognition*” is, itself, a metaphysical view (Rescorla, 2017). If cognitive science is guided by a computational *metaphysics* of mind, the questions it poses might be best answered in *metaphysically-committal* computational terms. But this seems to get things backwards: it is widely agreed that our metaphysics of mind should be *supported by* scientific work, not *constrain* it (e.g., Samuels, 2019). [↑](#footnote-ref-13)
14. Similar points seem to hold about the notion of computation in computer science and engineering, not just cognitive science. See Curtis-Trudel & Symons (n.d.), for a view in this area. [↑](#footnote-ref-14)
15. The capital letters refer to the response functions of different cone types, lower-case letters to weights, and *λ* to the incoming wavelength profile. [↑](#footnote-ref-15)
16. Current debates over computational perspectivalism have exactly this character: what is especially interesting about perspectivalism — the contribution of a researcher’s goals and resources to the project of computational explanation — is overlooked, and perspectivalism is rejected because computation “seems to be the wrong kind of thing to have its existence depend on mental states and/or social practices of sentient beings, or on the scientific perspectives they take” (Mollo 2021, 24). [↑](#footnote-ref-16)
17. As in Chalmers (1996); Milkowski (2013); Piccinini (2015); Brette (2022). [↑](#footnote-ref-17)