




The physicality of representation

Corey J. Maley¹ 

Received: 10 June 2021 / Accepted: 5 October 2021 / Published online: 11 November 2021

© The Author(s), under exclusive licence to Springer Nature B.V. 2021

Abstract

Representation is typically taken to be importantly separate from its physical implementation. This is exemplified in Marr’s three-level framework, widely cited and often adopted in neuroscience. However, the separation between representation and physical implementation is not a necessary feature of information-processing systems. In particular, when it comes to analog computational systems, Marr’s representational/algorithmic level and implementational level collapse into a single level. Insofar as analog computation is a better way of understanding neural computation than other notions, Marr’s three-level framework must then be amended into a two-level framework. However, far from being a problem or limitation, this sheds lights on how to understand physical media as being representational, but without a separate, medium-independent representational level.

1 Introduction

Discovering and understanding neural and mental representations is a central component of neuroscience and cognitive science. According to the generally-accepted view, investigating the brain does (and should) proceed via different levels. Marr’s three-level framework is extraordinarily influential here: Marr (1982) articulates the place of representation-based explanations as one particular level in the overall project of explaining any information-processing system, including the mind/brain. More specifically, Marr made explicit the idea that representations and algorithms (i.e. the representational/algorithmic level) can be studied and explained independently from “higher-level” investigations of the purpose and goals of the computations made possible by those representations and algorithms (i.e. the computational level). At the same time, representations and algorithms can be studied and explained independent

This article belongs to the topical collection “Interdisciplinary Perspectives on Locating Representations in the Brain”, edited by Jessey Wright and Sarah Robins.

✉ Corey J. Maley
cmaley@ku.edu

¹ University of Kansas, Lawrence, USA

of “lower-level” investigations of the specific physical media that implement those representations and algorithms (i.e. the implementational level). While a complete understanding and explanation of an information-processing system like a digital computer or the mind/brain requires all three levels, this three-way split of independent levels allows for a clear division of investigative labor, and perhaps a clear division of how the mind/brain is itself organized.

The proper understanding of Marr’s framework has generated much discussion, and many questions about its interpretation remain. One could take the framework to reflect ontological commitments, for example. On such an interpretation there would be some real sense in which information-processing systems have these different levels. Or one could take the framework to reflect explanatory or epistemological concerns. On that kind of interpretation the levels might only reflect the goals of particular investigative or explanatory practices. Recent research has focused on other topics as well, such as the importance and independence of each level (Bechtel and Shagrir, 2015), the levels at which one should posit mental content (Ritchie, 2019), and how to understand the representational/algorithmic level as connecting the computational and implementational levels (Love, 2015).

My goal here, however, is not to contribute to the literature on Marr per se. Rather, I will use Marr’s framework as something of a placeholder for what I take to be an important and long-lasting assumption in neuroscience and cognitive science, which is the independence of the representational-algorithmic level from the implementational level. Whether Marr’s framework *in particular* is the right way to understand these levels (or if they are best understood as “levels” at all) is immaterial: my target is the general view that representations and algorithms are, in some important sense, independent from the physical substrate(s) that implement them. This view—that representations and algorithms are independent of (or can be analyzed without reference to) their implementing physical media—has been referred to as “medium independence” in the philosophy of computation.

Now, when it comes to neuroscience, the assumption of medium independence may seem to imply that research on how to locate representations *in the brain* (the subject of this special issue) might be something of a category mistake if taken literally. Specifically, questions of what can be found at the neural level lie at the *implementational* level; questions regarding representations lie at the *representational* level, independent of the implementational level. Trying to find representations at the neural level might seem akin to looking for cursors or email programs in the electrical circuitry of a microchip: one will find the physical structures required to *implement* cursors and email programs, but one will not find cursors or email programs themselves.¹

So why is this *not* a category mistake? Certainly I (and many others, such as Thomson and Piccinini (2018)) do not take the search for representations in the brain to be misguided, or merely metaphorical. One way forward might be to further clarify how one understands Marr’s framework (or whatever other framework one might want to adopt). Another way forward—the one I will pursue here—is illuminated by ques-

¹ One should not push this too far, however. As an anonymous referee points out, it does not seem mistaken to take at least some physical objects as being representations, such as the punch cards of mid-twentieth-century digital computers, or even data recorded on a DVD.

tioning the independence of the implementational and the representational/algorithmic levels in the first place.

The idea is this: the view of the mind/brain as a computer—one of the foundational assumptions of cognitive science and the departure point for Marr’s levels—implicitly takes “computer” to mean “digital computer.” This, in turn, informs the idea that one can analyze representations and algorithms independent of their physical implementation. For digital representation and computation, this is a well-known and often repeated point: this type of representation and computation is medium-independent in a deep and important sense. However, the extent to which this independence applies to *analog* representation and computation is unclear; I will argue that it simply does not. In other words, the representational/algorithmic level is not separable from the implementational level the way they are for the digital case. This extends the implications for the idea that mind/brain is a computer without being a *digital* computer; instead it may be an analog computer, in whole or in part (Maley, 2018). To the extent that that is the case, it would then be a mistake to take the representational and implementational levels as separate and independent. For those who like slogans, we can say “No representation without implementation.” Or so I will argue.²

Here is the shape of things to come. First, I will review the Marrian framework, then outline the account of analog representation and computation I take to be most useful for neuroscience and cognitive science. Next, I will argue that, when it comes to analog representation in the brain and in analog computers, what counts as representational and what counts as implementational are not clearly distinct, especially when contrasted with the more familiar digital paradigm. This point will be made by examining cases of analog representation in analog computing machines and in neural systems. In each case, investigating the physical nature of the representational medium (i.e. the implementation) illuminates the nature of the representation itself. Thus, rather than a distinct, medium-independent representational level, these cases of representation are medium-*dependent*. As I hope to show, this has implications for how we view neural representation and computation, and “where” representations might be found in the brain.

2 Background: Marr’s levels, the analog, and the digital

This section will set the stage for the main thesis of this article by getting all of the elements on the table. First, I will briefly review the three levels of analysis originally presented in (Marr, 1982). This will include an illustration of how analyses at Marrian levels are orthogonal to other level schemata. For example, one can decompose the elements of a Marrian level analysis into smaller elements that themselves remain at the original Marrian level. Marr’s levels are simply *not* levels of abstraction or composition, a point that can lead to confusion. Next, I will review a recent account of analog computation and representation, pointing out the important differences between analog and digital computation and representation. On this account, the analog/digital

² “The Marrian representational/algorithmic level is actually a single representational/algorithmic/implementation level” would be more accurate, but less catchy, and a good slogan is nothing if not catchy.

distinction is *not* merely about continuity versus discreteness, as the received view of this distinction has it. Instead, analog representation is a fundamentally different way of representing that can be either continuous *or* discrete. What is most important for the thesis of this paper is that analog representation depends on the physical nature of its implementing substrate: it is *not* medium-independent in the way that non-analog representation is. We will return to that point in the next section.

2.1 Marrian levels

Like many sciences, neuroscience and computer science are committed to the view that there are levels to their objects of study.³ At the lowest resolution, computing machines have two separate levels: the hardware level and the software level. At a higher resolution, they have many more levels, including digital circuitry, registers and ALUs, machine code, assembly language, APIs, operating systems, and user interface elements (to name just a few). The same is true of brains: at a very low resolution, one might carve the brain up into individual lobes and their functions (broadly characterized). At a higher resolution, investigations might involve a particular area of cortex and its function; a still higher resolution might involve cortical columns, and so on. Different investigations take place at different levels, from the level of brain regions and networks to the level of ions and neurotransmitters.

Now, what precisely are these levels supposed to be levels *of*? The answer is not always clear, but even when it is, different answers serve different purposes. Scientists and philosophers of science have proposed various typologies, including levels of reality or being, levels of abstraction, levels of organization, levels of size, levels of mechanism, and many others (Craver, 2015). Here I will focus on Marr's levels, which Marr himself called levels of description or analysis (and sometimes known as Marr's "tri-level" framework or hypothesis). However, the points made here generalize to other level schemata that consider distinct levels of representation and implementation. In any case, because Marr's framework is rather well-known, I will only provide a brief overview here.

(Marr, 1982, p. 25) characterizes his three levels of the analysis of information processing systems, including cognitive systems, as follows:

Computational theory What is the goal of the computation, why is it appropriate, and what is the logic of the strategy by which it can be carried out?

Representation and algorithm How can this computational theory be implemented? In particular, what is the representation for the input and output, and what is the algorithm for the transformation?

Hardware implementation How can the representation and algorithm be realized physically?

³ It is particularly interesting that neuroscience and computer science share this feature, given that they grew up together. Many influential thinkers from the twentieth century explicitly related ideas about the workings of automata and neural machinery, including McCulloch and Pitts (1943) and von Neumann (1958). That history is itself fascinating, and in some ways foreshadows the advent of cognitive science, where the activity of the mind/brain is explicitly likened to computation (Lycan, 1990; Von Eckardt, 1993). I cannot discuss that history in any detail here, partially because a thorough chronicle has yet to be written. I simply note that parallels between neuroscience and computer science run deep.

To illustrate the idea, consider applying this framework to a simple pocket calculator. At the computational level (CL), this would involve specifying that, for the operation of addition, pairs of inputs (the summands) are to be mapped to a single output (the sum). For every mathematical operation required, we specify the appropriate mapping or function. At this level, we are not concerned with how numbers are represented, just that mathematical operations are performed in the requisite way. For simple operations like addition, there may not be much in the way of a “strategy” by which such operations are carried out at the computational level. However, for more complicated operations, we may have to specify a sequence of operations. For example, if we had a scientific calculator that can multiply polynomials or compute derivatives, we might specify the order of operations required to produce the requisite answers.

At the representational/algorithmic level (RAL), our calculator is analyzed in terms of digital representations (in the sense outlined in (Maley, 2011), and discussed in the next section). Briefly, digital representations of numbers involve numerals in specific places (i.e. digits), which just is the standard way that we represent numbers. Thus, three hundred fourteen is represented as the sequence of numerals “314”, where the value of the entire representation is a function of the numerals in their respective places. This type of representation lends itself to the application of particular algorithms. For addition, this is one of the standard techniques learned in elementary school: add the least-significant digits, then add the next-most-significant digits (plus the carry digit from the previous addition, if there is one), and so on. This kind of representation also constrains the kind of physical implementation possible. For instance, if the digital representation is in binary (i.e. base-2), then we need a physical medium that can be in one of two distinguishable physical states; if it is in decimal (i.e. base-10), then we need ten such states. However, the representational level is medium-independent in the sense that any physical medium can be used that satisfies these constraints.

Finally, at the implementational level (IL), we get into the details of the physical device used to implement the representations and algorithms. For a typical calculator that uses internal binary representations, this would involve a characterization of the circuitry used to store strings of digits, what voltage levels are used to represent the two states (e.g. the 0s and 1s of binary representations implemented as zero and five volts), the circuitry responsible for manipulating voltages in the correct order, and so on. This level is, by definition, medium-*dependent*: we must understand the physical properties of the media involved in order to understand how the device does what it does *qua* physical device.

For the purposes of this essay, it is particularly important to be clear about the relationship between representations and the three Marrian levels. We can summarize those relationships as follows:

Computational Level What is it that is represented? (e.g. proximal or distal stimuli, numbers, magnitudes, etc.)

Representational/algorithmic level What is the format of the representation? How do variations in the format correspond to variations in what is represented? How do algorithms manipulate those variations? (e.g. discrete or continuous, analog or non-analog, type of code or coding, etc.)

Implementational level How is the representation and algorithm realized physically? (e.g. electrical circuitry, chemical gradients, mechanical movement, etc.)

A particularly important observation is that the RAL does *not* specify *what* is represented; that is a computational-level specification. Instead, the RAL specifies *how* the things specified at the computational level are represented.

Besides being clear about these details, we should note an important source of potential confusion regarding Marrian levels, which is that they are orthogonal to analyses involving other level schemata. For example, given an analysis of an information-processing system at a particular Marrian level, the parts of that level might be decomposed into smaller units *without* thereby resulting in an analysis at a different Marrian level. Rather, such a decomposition would simply result in a higher-resolution analysis at the *same* Marrian level. This point is particularly important for later discussion, so let us look at a concrete example.

Returning again to the example of the pocket calculator, let us focus on an analysis at the RAL. While the internal representations are binary (which is simpler and more efficient to physically implement than decimal), the input and output is displayed in decimal (which is simpler for humans to understand by convention). Consider just the output. The calculator will conclude an operation with a number internally represented in binary, which then needs to be converted to decimal in order to be displayed. That is not quite accurate however: we do not really need to *convert* the number to a decimal format, but only *display* its digits in decimal format. Thus, if the output of a calculation is five, then we will have an internal binary representation of 0101, which will need to display the numeral 5. A schematic of this analysis is diagrammed in the top half of Fig. 1. At this level of detail, we only characterize the fact that our internal, binary representation must be transformed (in some unspecified way) to the numeral 5.

We may want more detail about these representations, but not about the *implementation* of these representation. We can characterize what happens at this stage at a higher resolution while still remaining at the RAL. For example, if we want to use a seven-segmented display to show our decimal digits (a common strategy in pocket calculators), then we can analyze the transformations that need to occur at the level of individual bits. This is depicted in the lower half of Fig. 1, where a truth table shows how the individual bits of the output are mapped to the activation of the individual segments of the display. This is *not* an analysis at the IL, because we are not yet concerned with the physical devices that will implement this part of the computation. Rather, we are still dealing with representations (binary digits) and algorithms (truth-table mappings), but at a higher level of detail.

Another way to put this point is that the low-resolution analysis (the top half of Fig. 1) is an abstraction of the high-resolution analysis. On the received view of abstraction and idealization (Potochnik, 2017), abstraction is the process whereby certain details of an explanation or analysis are left out in order to focus on others (e.g. leaving out details about material properties, except for mass, in a simulation), whereas idealization is including in an analysis details which are known to be false, but which make relevant explanatory details clearer (e.g. positing infinite population sizes, or frictionless surfaces). There is clearly much more detail at the high-resolution analysis (e.g. details of the particular transformation given in the truth table), but much

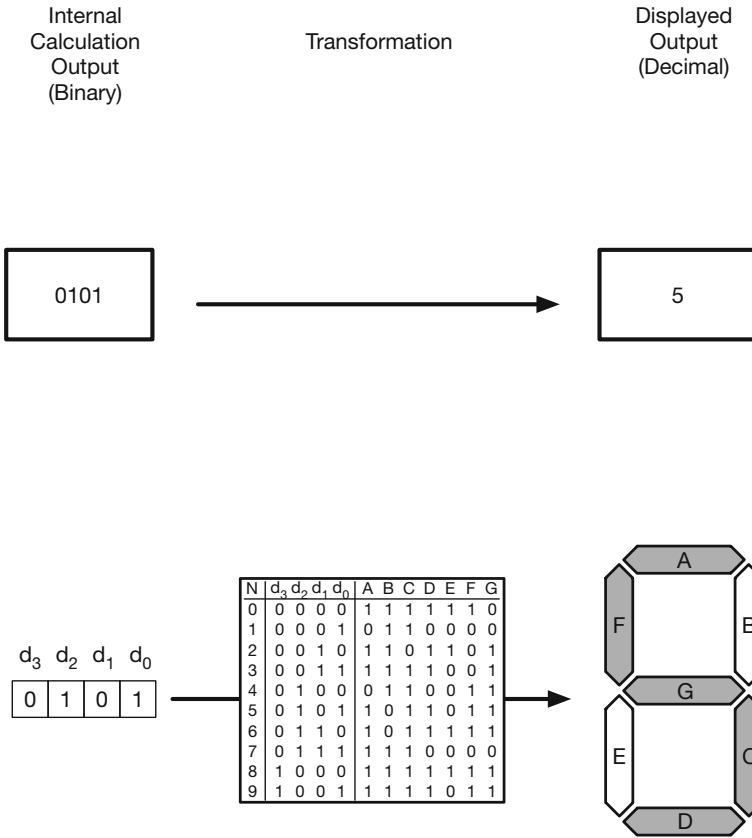


Fig. 1 Top: schematic of conversion of output from internal format to display format. Bottom: higher resolution decomposition of the same conversion

of it may be distracting for certain explanatory purposes, or even limiting to other applications (such as when the “5” is displayed in some other way). Most important to note, though, is that the relationship between the two parts of Fig. 1 is not one of implementation: they are both analyses at the RAL level, albeit at different resolutions.

A further decomposition (i.e. a still higher resolution analysis) may not be possible at this Marrian level, simply because we are at the level of individual binary digits and the individual segments that compose the output display. However, this is not surprising or problematic: analyses have to bottom out somewhere. On the other hand, we could switch to an IL analysis, showing how the bits are physically implemented, and illustrating the physical configuration of the circuits that would implement the transformation characterized by the truth table. But that would *not* be a further decomposition at the RAL ; that would be moving to a different Marrian level entirely.

2.2 Analog and digital representation

According to the received view of the analog/digital distinction, “analog” is synonymous with “continuous,” and “digital” is synonymous with “discrete.” On this characterization, the feature that divides one from the other is simply whether the representations involved vary smoothly or in discrete steps. Because representations are either analog (i.e. continuous) or digital (i.e. discrete) on this view, an account of one automatically gives you an account of the other. They are mutually exclusive and jointly exhaustive.

However, there is an alternative way to characterize analog and digital representation that is more useful for neuroscience, cognitive science, and computer science (at least insofar as it correctly classifies analog computing machines of the twentieth century). This alternative is what Adams (2019) calls the “Lewis-Maley” account of analog and digital representation (Lewis, 1971; Maley, 2011). On this account, the difference between analog and digital representation is not as simple as discrete versus continuous; instead, they are quite different in kind. As such, a characterization of what counts as analog does not automatically generate an account of what counts as digital (or vice versa): each piece of the “distinction” needs its own, separate characterization. In brief, analog representation is the representation of numbers via physical magnitudes, while digital representation is the representation of numbers via their digits. What follows will be a brief overview of the Lewis-Maley view. More details can be found in (Maley, 2011).

2.2.1 Analog representation

Analog representation, according to the Lewis-Maley view, is a species of what Beck (2018) calls “mirroring” accounts. These accounts characterize analog representation as involving a kind of mirroring between the representation and what is represented; they are the “representation of magnitudes, by magnitudes,” (Peacocke, 2019, p. 52). For example, a mercury thermometer is analog because the height of the mercury represents the temperature; thus, an increase/decrease in the mercury’s height reflects an increase/decrease in the temperature represented. Similarly, the angle of the second hand of an analog clock represents the number of seconds past the minute.

This point is worth belaboring a bit. On this account, analog representation is characterized as an analogy (hence, “analog”) between the physical representation and what is being represented. Besides the examples just given, an hourglass is an analog representation of elapsed time, and a vinyl record is an analog representation of sound. In each case, a physical property of the representation—the *very property doing the representing*—covaries with what it represents. Again, the increasing/decreasing height of the liquid in the thermometer represents a corresponding increase/decrease in temperature. Similarly for the sand in the hourglass with respect to time, and the ridges and grooves in the vinyl with respect to the frequencies and amplitudes of sounds.

This is clearer when contrasted with digital representations. In a digital thermometer, numerals (or strings of numerals) represent a number, and the magnitude of *that* number represents an increase/decrease in temperature. However, there is no *literal* increase in the representation itself. Thus, the two-digit sequence “54” is no smaller

or larger than the sequence “55,” even though fifty four (the number represented by “54”) is less than fifty five. However, a 54 mm column of mercury is *literally* shorter than a 55 mm column (more on this point is in the next section).

Importantly, in this characterization of analog representation, the quantities above can be continuous *or* discrete. Take the hourglass, for example. The representational medium—the “sand”—could be a continuous fluid, or a true particulate substance. The analog representation is there in either case: the more substance there is (fluid or number of particles), the more time has passed. Furthermore, if the substance is discrete, it also does not matter if the relevant kinematics are continuous or discrete. Suppose one grain of sand falls every second.⁴ We still have an analog representation. There is no essential difference, *qua* analog representation, whether the quantity being represented, or the quantity doing the representing, is continuous or discrete.

Now, when it comes to analog computation, analog computers often used continuous representations. However, there are also examples of analog computers that used *discrete* (rather than continuous) representations and processes. One simple example is a discontinuous (i.e. discrete) programmable step function generator: while still analog, the representations used are *not* continuous (Maley, forthcoming). Whereas the received view cannot make sense of what is *analog* about these analog computers, the Lewis-Maley view can: they are simply computers that use discrete analog representations, rather than continuous ones. In fact, it is the same principle involved in an analog watch that ticks in discrete steps. It is analog in virtue of the fact that it is doing its representing via a physical magnitude (in this case, the angle of the hands); it does not matter whether the variation occurs smoothly or in steps. Analog clocks do not turn into digital clocks when they tick instead of running smoothly.

2.2.2 Digital representation

As was the case for analog representation, the account of digital representation most useful for our purposes, which include understanding computation, was originally suggested by Lewis (1971), then further defended and extended by Maley (2011). On this view, digital representation is representation of a number by its digits, where digits are understood as numerals in specific places; as mentioned above, this is, in fact, the way we typically represent numbers.⁵ Thus, the digital representation of the number five hundred twenty five and three tenths is the four-digit (plus one decimal point⁶) sequence “525.3”, which has the value

$$(5 \times 10^2) + (2 \times 10^1) + (5 \times 10^0) + (3 \times 10^{-1}) = 500 + 20 + 5 + .3 = 525.3$$

⁴ Thanks to Felipe De Brigard for this example.

⁵ The fact that this kind of representation is so common makes pointing out its peculiarities all the more important. Johnstone (2006, p. 1) said it best: “It is, after all, the familiar that is so strange.”

⁶ The use of ellipses in the expressions that follow obscures the decimal point; however, the decimal point is necessary for the digital representation of rational and real numbers.

More generally, for base b and digits $d_k \dots d_0, d_{-1} \dots d_{-m}$, where each digit $d_j \in \{0, 1, \dots, (b-1)\}$, the digital representation “ $d_k d_{k-1} \dots d_0 d_{-1} \dots d_{-m}$ ” has the value

$$(d_k \times b^k) + (d_{k-1} \times b^{k-1}) + \dots + (d_0 \times b^0) + (d_{-1} \times b^{-1}) + \dots (d_{-m} \times b^{-m})$$

Note that in order to represent rational and real numbers (and for the decimal point to be meaningful), the base b must be two or larger.

Why is this particular way of understanding “digital” useful for understanding computation? Simply because contemporary digital computers are not only discrete, but digital in exactly the sense just articulated. Data and instructions in digital computers are not just strings of discrete symbols, but strings of discrete symbols that represent numbers in the usual binary way. Manipulations of these representations depend crucially on interpreting them as numbers, with more and less significant digits. For example, different computer architectures use different “endianness,” which is the order in which bytes are stored. Big-endian computers store bytes the way we typically write numbers: the most-significant element comes first (i.e. the leftmost), and the least-significant element comes last (i.e. rightmost). Little-endian computers are the reverse, similar to how we could write numbers with the most-significant digit on the right. This choice is arbitrary, and different computers use different conventions. However, *within* any particular computer, knowing which digit is most significant is crucial.⁷

In marked contrast to the received view, characterizing analog and digital representation in this way results in a view according to which these two types are *not* jointly exhaustive: they do not carve the entire space of possible representations into two clean sectors. For example, symbols such as π and words such as “desk” are neither analog nor digital, even though each is discrete in a straightforward way. Furthermore, many non-representational yet continuous phenomena are not analog (because they are not representations per se), such as liquids or electric current. One could, of course, use quantities of liquid or current as analog representations (as they are in real-world examples of analog computers). But keeping “continuous” and “analog” distinct avoids confusing things that really are representations for things that are not.

2.3 First- and second-order representation

Proponents of the received view would agree that analog and digital representation are fundamentally different (after all, continuity and discreteness are mutually exclusive). I mentioned above that this distinction is not correct, however, because of the *discrete* analog representations found in analog computers. The deeper distinction between analog and digital representation that needs to be made clear is that analog representation is a kind of first-order representation (i.e. the representation of the magnitude of a number by physical magnitudes), whereas digital representation is a kind of second-order representation (i.e. the numerical representation of a number by its digits, which themselves are individually represented by variations in the values of a physical prop-

⁷ Maley (2011) goes into much more detail about why digital computers are digital on this understanding of the term.

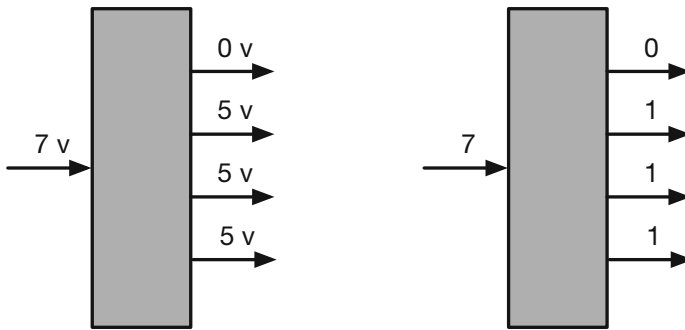


Fig. 2 Left: physical characterization of input and output lines. Right: what those physical values represent. Note that this is not a Marrian-level analysis

erty). There is much to unpack here, but doing so will help us understand precisely how analog and digital representation and computation differ (as well as what the received view of the analog/digital distinction misses).

Consider the example of a four-bit analog-to-digital converter, adapted from section 4.1 of (Maley, forthcoming), shown in Fig. 2. This converter has one input and four outputs, and its job is to convert an analog representation of a number into the digital representation of the same number (all within the context of an electronic computer). In this example, we are converting an analog representation of the number seven into its binary digital representation. I am using the example of a converter here simply to illustrate the difference between analog and digital representations side-by-side. Also, this is decidedly *not* a Marrian analysis; that point will be important later.

On the left side of Fig. 2, this device is characterized only in terms of its physical inputs and outputs. When the input is seven volts, then the first (top) output will be zero volts and the next three will each be five volts. Characterized purely physically, this device maps single voltages to patterns (i.e. ordered quadruples, or sequences) of voltages.

On the right side of Fig. 2, this device is characterized in terms of what the voltages represent. The input—the analog representation—represents the number seven (by seven volts). As for the outputs, each one represents either the numeral 0 (by zero volts) or 1 (by five volts); thus, the four outputs, taken together, represent the sequence “0111.”

Importantly, whereas the analog representation requires no further interpretation, the digital representation does. The sequence “0111” *per se* is not what we want to represent; rather, this is a sequence of numerals in specific places (i.e. digits). Furthermore, given that we already stipulated that this is a *binary* analog-to-digital converter, and given the assumption that we know which digit is the most-significant (i.e. the top-most), we know that the correct interpretation of this sequence of digits is $(1 \times 2^2) + (1 \times 2^1) + (1 \times 2^0) = 4 + 2 + 1 = 7$.

Now, exactly *how* this device converts the analog representation to the digital representation does not matter here. This example is only meant to show that analog and digital representations do their representational work in very different ways. Given a number X , an analog representation of X is a representation of the *magnitude* of X via

the magnitude of some physical quantity. A digital representation of X , on the other hand, is a representation of the *digits* of X , where those digits, in turn, are represented via distinguishable values of a physical property.

This point can be put in a different way. Whatever numbers themselves *are*, they are, in some sense or another, abstract. Thus, one cannot really manipulate numbers at all, except indirectly via concrete representations of them. One way to represent them is to represent their magnitude, and one way to represent their magnitudes is with other magnitudes. However, because we are talking about concrete representation, we must represent the magnitudes of numbers with *physical* magnitudes. And a straightforward way to represent magnitudes with physical magnitudes is to use a homomorphism between the *represented* magnitude and the *representing* physical magnitude. This gives us analog representation where, as mentioned above, an increase (or decrease) in the relevant physical magnitude (e.g. the height of the liquid in the thermometer) represents an increase (or decrease) in the magnitude being represented.

Another way to represent numbers is to represent them by their digits.⁸ In this case, sets of physical properties represent sequences of digits, where those digits, in turn, represent the number. I say “physical properties” instead of “physical magnitudes” because, unlike analog representation, digital representation only requires distinguishable differences in some property or another, and there need not be a homomorphic relationship between the numerals represented and what is doing the representing. One could represent the decimal (base-10) digits of a number using arbitrarily-assigned colored squares for each numeral, for example.

There are many others ways that numbers can be (and have been) represented, such as purely conventional symbols like π and e , or the use of cumulative systems like Roman numerals; analog and digital representation are far from the only games in town (Chrisomalis (2020) provides a fascinating history and analysis of a variety of such systems). However, analog and digital representation (properly understood) are the most relevant for understanding the role of representation in information processing systems, so we will not consider other schemes here.

In summary, analog representation is a kind of first-order representation: a physical property (in this case a magnitude) represents the magnitude of a number. Digital representation, on the other hand, is a kind of second-order representation: a physical property (not necessarily a magnitude, but some property with distinguishable values) represents numerals, and the numerals, interpreted in the right way (i.e. according to their order and the base), then represent a number.

So much for getting the relevant background concepts on the table. Let us now turn to why it was important to do so.

⁸ There is, of course, one way in which the digits of numbers *do* represent their magnitude, albeit in a rather different way than what concerns us here. This is hinted at in the way that we interpret the digits of a number: the leftmost digit is the most significant, the rightmost is the least significant, and each numeral has a different interpretation, depending on which digital place it is in. However, for our purposes, we will set this complexity aside, simply because it will take us too far away from the main thesis of this essay.

3 Analog physicality and the Marrian collapse

The idea that I propose here is that, when it comes to analog computational systems (which relies on analog representation), Marr's levels must be applied in an amended form such that the representational/algorithmic level (RAL) and implementational level (IL) collapse into a single level. This may be interesting in its own right, but is particularly important for cognitive science and neuroscience, simply because (as I will argue in the next section) it is more likely that neural systems are analog computational systems, rather than digital (Maley, 2018). First though, we will look in more detail why analog representation and representation requires reference to the implementing medium (and is thus medium-*dependent*, contra accepted views about the necessity of medium-*independence* for computation (Piccinini, 2020), leading to the Marrian collapse.

3.1 Analog physicality

Characterizing analog representation *qua* representation requires reference to the physical details of the system that implements those representations in a way that digital representation does not. This is due to the fact that, with respect to their physical implementation, analog representations are what I above called first-order, whereas digital representations (and non-analog representations more generally) are second-order. The idea I develop here is that analog representation simply cannot be separated from its physical implementation in the way that digital representation can. Thus, in terms of the Marrian framework, the RAL and the IL are one and the same.

Consider the following initial value problem (a differential equation with initial conditions, adapted from (Peterson, 1967, p. 36), cited in Maley (forthcoming)):

$$y = x'' + 3x' + 16x; \quad y = -80$$

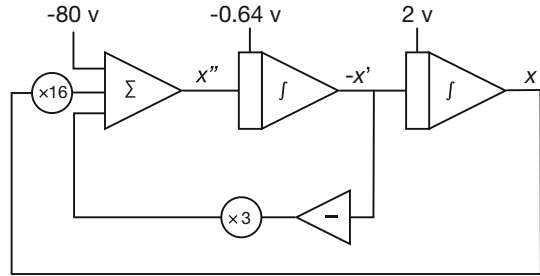
$$x'(0) = -0.64; \quad x(0) = 2$$

An electronic differential analyzer (EDA), one of the oft-used components in many electronic analog computers, can be set up or programmed to solve this system. Doing so requires that we arrange certain electronic components in a way that reflects the mathematical structure of this system. In this example, shown in Fig. 3, we have an adder (triangle with Σ), two integrators (triangles with \int), a sign inverter (triangle with $-$), and two multipliers (circles with \times).

Notice that this seems to be an IL diagram: the inputs and outputs are all voltages, and the adder, integrator, and inverter are all electrical components. We could, of course, examine these implementational level elements at a higher resolution: a single integrator, for example, is itself composed of several individual electrical components. But that would only be increasing the resolution of what is already an analysis at the IL.

Now, given this IL analysis, what does this same set of components look like in a RAL analysis? Much like the example illustrated in Fig. 2, it might seem that we only need to remove the voltage signs for the variables, because the number of volts

Fig. 3 Electronic analog computer schematic



represents the magnitude of the value of the variables. Given the initial condition that $y = -80$, we represent the value -80 by -80 volts; similarly for the other variables and their values.⁹ As for the individual components, what they represent is given by their names. The integrator (the component with the \int) represents integration, the adder represents addition, and so on.

However, this is not quite right. Recall that Fig. 2 is *not* an illustration of Marrian levels, and that at the RAL, we do not find what *is represented*; that is for the CL. Instead, at the RAL, we find what *does the representing*. While it is true that -80 volts represents the value -80 , -80 is *not* a representation: it is the thing represented! So what exactly *are* the representations of the RAL in this system?

If we attend closely, it is not quite correct to say that is an IL characterization of this analog computer. Instead, this example is a *hybrid* representational/algorithmic/implementational level characterization (which is very often the case when it comes to these kinds of analog computer diagrams). Strictly speaking, addition and integration are *mathematical* operations, not defined on physical quantities. Strictly speaking, it does not make sense to apply a mathematical operation (addition) to a physical property, such as the voltage level of a particular circuit element. However, we also have a clear idea of what the physical interpretation of addition must be when applied to voltages: it requires producing an electrical potential of $(x + y)$ volts when given two separate electrical potentials x and y as input. Similarly, the “multiplication” of two voltages with potentials x volts and y volts produces a voltage of $(x \times y)$ volts.

This is another point worth belaboring. The ways that we map mathematical operations to physical operations is often commonplace, which obscures some interesting facts about them.¹⁰ For many physical phenomena, the mapping is more-or-less obvious. In the case of length, physical “addition” amounts to physical concatenation: “adding” two lengths just is the operation of putting one length next to the other. “Multiplication” amounts to repeated addition of lengths. As such, it makes sense to say of one length that it is the sum of two others, or that a length is three times as long as another. For other physical properties, these mappings are less straightforward for certain mathematical operations. Temperatures (in degrees Fahrenheit, say) can

⁹ In real electronic analog computers, these values would sometimes need to be scaled by a linear factor so as to keep the values within reasonable ranges for the machine in question: dealing with millions of volts may well overload the system, not to mention pose a serious danger for the operator.

¹⁰ A recurring theme! See footnote 5.

be added in a straightforward way; the temperature difference between 40° and 35° is the same as the difference between 35° and 30° . However, temperature does not admit of a straightforward multiplicative operation that can be physically interpreted. It would be confused, for example, for one to understand 26° as the temperature one produces when one heats up 2° by a factor of 13: in other words, 26° is not thirteen times as warm as 2° (which itself is not infinitely many times as warm as 0°). In the electronic analog computer, the voltages used are like length in that they admit of physical operations that can be straightforwardly mapped to mathematical operations in an obvious way.¹¹

Going back to the case of the EDA, the multipliers, integrators, and so on all straddle the line between the IL and the RAL. This is not simply a fluke of this particular diagram or analysis; this is the defining feature of analog computation and representation. It is only possible to characterize these components in terms that are both implementational (i.e. they make essential reference to the relevant physical properties) and that are representational (i.e. those physical properties are the very properties that do the representing). These are devices that traffic in electrical properties, and the particular trafficking they are engaged in is understood in representational terms.

The point becomes clearer still when we examine the specifically *algorithmic* aspect of an RAL analysis. Consider the operation of mathematical integration. In the case of a digital computer, a particular algorithm would have to be used for the integration of the relevant function. Examples include a simple piecewise-trapezoidal approximation, or (more likely) the Runge-Kutta algorithm. Each of these algorithms gives a numerical approximation to the definite integral of a function f over time for a given interval.¹² Given an initial condition y_0 at time t_0 with step-size h , the Runge-Kutta algorithm for the value of y_{n+1} for each consecutive time-step t_{n+1} is given by:

$$\begin{aligned}k_1 &= f(t_n, y_n) \\k_2 &= f\left(t_n + \frac{h}{2}, y_n + h\frac{k_1}{2}\right) \\k_3 &= f\left(t_n + \frac{h}{2}, y_n + h\frac{k_2}{2}\right) \\k_4 &= f(t_n + h, y_n + hk_3) \\y_{n+1} &= y_n + \frac{1}{6}h(k_1 + 2k_2 + 2k_3 + k_4) \\t_{n+1} &= t_n + h\end{aligned}$$

Contrast this with integration that takes place in the EDA, part of the electronic analog computer shown in Fig. 3. Here, integration does not use an algorithm, at

¹¹ In technical terms, voltages and lengths form a ratio scale, whereas temperature only forms an interval scale (Stevens, 1946). Physically (and roughly) speaking, a length of zero corresponds to a magnitude of zero, whereas a temperature of zero is arbitrary (hence the different zero points in the Celsius and Fahrenheit scales). Much more on the metaphysics of these issues can be found in (Wolff, 2020).

¹² The independent variable need not be time, but by convention it is often used in the explication of these algorithms.

least not in anything like the usual sense. There is simply the manipulation of the voltages. Of course, this manipulation corresponds to the mathematical operation of integration. But the way in which the component does this operation is not specifiable without reference to the physical media of the device, namely, voltages in this case. The same point applies to the other components, such as the multipliers and adders. They all manipulate voltages in a way that corresponds to a natural mapping between mathematical operations and physical manipulations (which is, after all, how the components are named). This is in contrast to the components in a digital computer, where the algorithm might operate similar to how children sometimes learn to add (and multiply) numbers in elementary school: first the least-significant (right-most) digits are added, then the next-most-significant, etc.

Now, it may be tempting to think that the difference here is one of *following a rule* versus *acting in accordance with a rule*. Specifically, one might think that the digital computer performs its task by following a rule—in this case, the Runge-Kutta algorithm for integration, which would be specifically programmed. At the same time, one might think that the analog computer performs its task by merely acting in accordance with a rule: there is no rule that the analog computer is following explicitly, although what it happens to be doing is specifiable by a rule. However, this thought misses exactly what is happening in the analog case. It is not just that the analog computer is merely acting in accordance with a rule, like the Runge-Kutta algorithm: there is no rule at all. Instead, there is just a physical transformation that can be straightforwardly mapped to a *mathematical* operation, and mathematical operations are not rules (even though that is how we very frequently perform those operations).

These points can be summarized as follows: digital representation abstracts away from its physical implementation, whereas analog representation takes advantage of it essentially; we began to see this point in Sect. 2.3. Digital representation is medium-independent, whereas analog representation is medium-*dependent*. In the digital case, the numbers are represented via a representation of their digits, thus manipulation of the numbers requires the manipulation of the individual digits. But in the analog case, numbers are represented via a representation of their magnitudes, and manipulation of the numbers requires directly manipulating those magnitudes. Different physical media will use different physical manipulations, even if those manipulations can be mapped onto the same mathematical transformations. An electronic analog adder, for example, could work by adding its voltages by having its elements in a series: voltages in a series are such that the total voltage is the sum of the individual elements. A mechanical analog adder, however, could work by an arrangement of spur and differential gears, where the rotation of an output shaft is the sum of the rotations of input shafts. Again, these are not algorithms, much less different algorithms: they are simply different physical manipulations onto which the same mathematical operation can be mapped. Let us look at these differences more closely, and how they support the Marrian collapse thesis.

3.2 Collapsing the representational/algorithmic and the implementational levels

Given the considerations from the previous section, it may seem tempting to say that analog computers do not have anything that would correspond to an RAL analysis at all, which, in turn, would be to deny that they have or manipulate representations. However, I argued earlier that they *do* have representations of a specific kind: analog representations. Thus, it is not quite correct to say that they lack an RAL altogether. Instead, the RAL and the IL are one and the same when it comes to analog computers. We began to see what that means in the previous section, so here we will approach this view from another angle.

In a typical Marrian analysis, the IL and the RAL are taken to be independent. This is clear in the case of digital computers. One can understand, explain, and create the electrical circuitry of a digital computer (or whatever the physical substrate of the implementation happens to be) without any consideration of the possible programs that will be run on that hardware. Alternatively, one can understand, explain, and create complex software programs without any consideration of the hardware that those programs will run on. In short, there is a hardware level (the IL), there is a software (the RAL), there is a task level (the CL), and they are all independent from one another.¹³

This independence, however, should be not overstated; the levels constrain and influence one another, albeit loosely. The IL must have the right kinds of elements and properties to implement the elements of the RAL level. A digital system typically uses binary representations, for example, so it must have IL elements that can vary along some dimension in a way that allows for two distinct states. Plus, the real-time performance of a system will be constrained by the speed of the basic operations of the IL level. Different algorithms implemented in the same hardware an result in differential performance, but the same algorithm implemented in different hardware can also result in differential performance. Nevertheless, an analysis at the IL is independent of one at the RAL level, and the physical elements of the IL are not relevant to elements of the RAL. Furthermore, CL considerations might constrain the kinds of representations and algorithms the system uses, given the tasks the system is meant to perform. If a task requires large lists of numbers, certain data structures will be used; if it requires the manipulation of large quantities of text, others will be used. Nevertheless, CL analyses are independent of those at the RAL and IL.

The Marrian analysis of an analog computational system (as well as neural systems, as I argue in the next section) proceeds differently than that of a digital system. Before we look at differences, however, let us look at similarities, which can be found at the CL. Consider a task that requires mathematical integration. Conceptually, integration is much like calculating the running total of a function over time. For example, if we can measure the speed of an object as it moves, we can integrate that speed as the speed changes to determine the object's overall position. This turns out to be useful for a great many problems, both in real and artificial systems. Thus, specifying that mathematical integration is required is an element of a CL. Once again, we are not

¹³ Different researchers may want to characterize these levels differently (i.e. hardware, software, and task), but that is beside the point here.

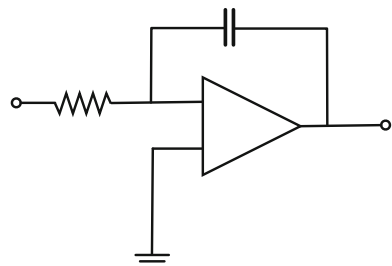
concerned at the CL with the kind of representations or algorithms used, nor how they are implemented; we are only concerned with what is represented, the overall computational task, its logic, and its appropriateness.

Things change when we look at the RAL level. As we already saw above, when it comes to the digital system, we will have to specify that we are using digital representations, and what particular algorithm to use for integration. Previously, we used the Runge-Kutta algorithm as an example. However, this algorithm was given in a rather low resolution RAL analysis. That algorithm did not specify how basic operations like multiplication, division, and addition are accomplished; they were taken for granted. But we may want a higher-resolution analysis which does specify how these basic operations are accomplished. For example, it was mentioned that addition is performed digit-by-digit in a manner nearly identical to how we normally add numbers by hand. Multiplication and division are more complicated, but again, they are performed by manipulating the individual digits of the digitally-represented numbers. Thus, a higher-resolution RAL analysis would include details of how these operations are performed on the digital representations in question (which will vary depend on the type of digital representation, such as the number of digits stored, whether the representation is in base 2, 10, or something else, etc.). And again, this would still be an RAL analysis because we are not yet concerned with any of the physical details of the implementing hardware.

As for an analog system, it is the physical properties of the system that do the representing. We have seen this a couple times now, but the point bears repeating. Consider again the elements from Fig. 3. Here, the voltages represent the values of this particular system, so the voltages are the representations. As for the algorithmic aspect of the RAL, we have only treated the integrator as something of a black box, so let us increase our resolution and look at this component in more detail.

The electronic integrator shown in Fig. 4 produces an output that is a function of the starting value of the output, the input, and how long the input has been present. It does this via the feedback loop between the output and the input. This design is such that the output just is the definite integral of the input. The electrical theory of why this is true is beyond the scope of this paper, but it is a basic consequence of this circuit design and its elements. Notice that, as before, we did not move to a different Marrian level, but only looked more closely at the RAL level. As before, the inputs and outputs of this integrator are still representations.

Fig. 4 Electronic integrator, composed of an operational amplifier with feedback between the output and input



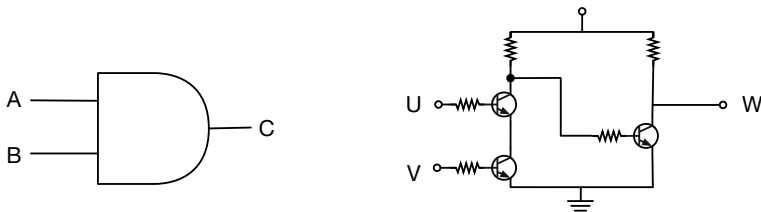


Fig. 5 Left: digital adder at the representational/algorithmic level. Right: physical implementation of the digital adder

Finally, let us look at the IL level, first of the digital system. This level includes the specification of the physical circuitry involved in the implementation of particular RAL elements. For example, a high resolution RAL analysis might include individual AND gates, which produce as output the logical conjunction of their two inputs.¹⁴ On the left side of Fig. 5, we have an RAL analysis of this component; the inputs and the output (labeled “A,” “B,” and “C”) all take the numeric value 0 or 1, being the atomic elements (i.e. numerals) of the binary digital representations the system uses. The right side of Fig. 5 shows the IL analysis of this same component. Here, the inputs and the output (labeled “U,” “V,” and “W”) are no longer individual numerals, but physical quantities—specifically, voltages that range from 0 volts to 5 volts. The individual components of the AND gate consists of resistors and three transistors. The circuit of which this system is a part is designed such that the circuits are generally bi-stable, meaning that most circuit elements transition quickly between 0 and 5 V, or from 5 to 0 V. Thus, the voltages of the implementational-level (0 and 5 V) map to the numerals of the representational/algorithmic level (0 and 1).

What does the IL level look like for the analog system? It is exactly the same as the RAL level. Because the physical properties represent the values of the system in a first-order manner (as discussed in Sect. 2.3), the RAL level and the IL level are the same level. There are no further details that can be added to the RAL level we just examined. As always, we can look at this level at a higher or lower resolution; but we would still be looking at the implementational details, which are also the representational/algorithmic details.

One might object at this point that we are relying too heavily on the particular details of this analog electronic computer. After all, given the set of equations above, we could use instead a purely mechanical analog computer that does use any electronic components at all. Or, we could use an analog computer like the MONIAC, which uses fluid flow (Isaac, 2018). Why could we not specify the RAL more abstractly than that given in the circuit diagram of Fig. 3, where we simply have abstract values, rather than voltages in particular? Then, one might think, we would have an RAL that is *not* committed to any particular implementational details, and thus could be implemented by any one of the different types of analog computer just mentioned. Voilà! The RAL and IL are separate after all.

¹⁴ In a binary digital computer we typically treat the two values as the numerals 0 and 1 instead of the logical values true and false, so strictly speaking the output is not logical conjunction but Boolean addition. That detail is unnecessary here.

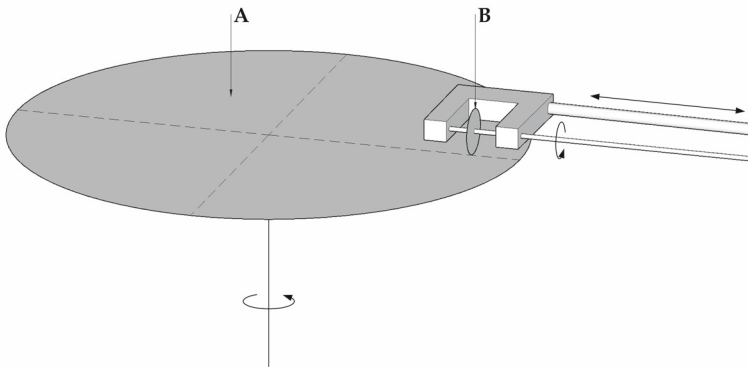


Fig. 6 A mechanical integrator. The left-right displacement of B is the input, and the output is the running total number of rotations of the connected shaft

The problem with this objection is that it confuses the RAL with what the elements of that level *represent*; a point mentioned earlier. It is true that, for example, 80 volts represents 80. However, if we abstract away from this particular way of representing 80, then we are simply no longer involved in providing an analysis at the RAL level. Instead, we have moved to the CL, where the elements are the things that are represented. One cannot have an RAL analysis that does not specify *how* things are being represented. But a CL analysis is agnostic as to what kind of system is being used or what kinds of representations it is using—we saw earlier that an analog system and a digital system can have the exact same CL analysis. Thus, it is not surprising that two different types of analog system might have the same CL analyses, too.

Once again, the reply to this objection is made clearer by looking at the algorithmic aspect of the RAL. Different types of physical processes accomplish transformations on representations in different ways for different types of analog systems. The way integration is performed in the electronic analog computer is via the operational amplifier just mentioned. But in a mechanical analog computer, integration might be performed by a disk integrator, shown in Fig. 6 (adapted from Maley, forthcoming).

This particular process for integrating a function relies on the physical displacement of the disk B (the input) relative to the center of disk A, which rotates at a constant speed; the output is the running total number of rotations of the shaft driven by disk B (see Maley, forthcoming for details). It does not rely on electronic components, and is unlike the working of the electronic integrator in any straightforward way (for example, the electronic operational amplifier has a feedback component, whereas the disk integrator does not). The only thing that unites the electronic and the mechanical integrators is that they both perform mathematical integration, which is a CL specification. Thus, an RAL analysis of a mechanical analog computer and an electronic analog computer will have to specify the different physical properties that are responsible for representations, and the different physical processes that are responsible for the transformation of those representations. However, because all of this necessarily requires reference to *physical* properties and processes, the RAL and the IL are one and the same.

A final point must be made before concluding this section. We should be careful not to overstate this necessary reference to the physical. After all, both of the integrators—the mechanical and the electronic—can be multiply realized (Piccinini and Maley, 2014; Polger and Shapiro, 2016). In other words, the mechanical integrator can be made of many solid materials, such as steel, aluminum, or even oak; the electronic integrator could use many conductive materials, such as copper, steel, or gold; the MONIAC computer could use different types of fluids. However, the particular material only matters insofar as it is able to instantiate the relevant physical mechanism. To be sure, some do consider these differences in material alone to count as multiple realizations (e.g. (Aizawa and Gillett, 2009)). For our purposes though, it is enough to note that differences in material properties that do not have an effect on the mechanistic properties of the IL do not count as multiple realizations of computational types. After all, we do not distinguish different types of, say, an AND logic gate by whether it uses germanium, silicon, or gallium arsenide as its primary semiconductor material, but by the mechanistic design of the electrical circuit.

4 The physicality of neural representation

So far, I have argued that analog representation and computation is fundamentally different than digital representation and computation because it uses physical properties directly (what I called first-order physical representation) as opposed to indirectly (what I called second-order physical representation). Then, I argued that, as a consequence of this, the Marrian RAL and IL collapse into a single level, but a single level that is simultaneously representational/algorithmic *and* implementational (in other words, it is *not* the case that either of these Marrian levels is eliminated). In this section, I will illustrate what this has to do with neural representations. I will expand on some of the ideas already presented in (Maley, 2018), which argued that at least some neural computation is best characterized as analog computation in the sense argued for in this essay. However, I will also show that the idea that Marr's RAL and IL collapse into a single level is already present in some recent work in neuroscience.

One thing worth making explicit at this point is that I do not mean to claim that all neural processes will (or must) fit into the analog computational framework I have outlined here. After all, not all research in neuroscience fits into a computational framework more generally. However, insofar as neuroscientists do understand neural processes as computational, those computational processes are best understood as analog, according to the framework offered here. Thus, if one does want to adopt the Marrian framework, the inseparability of the RAL and IL applies.

4.1 Neural coding

Neuroscientists have discovered a number of ways that neurons represent stimuli. There are more than can be covered here, but I will highlight a few that illustrate the

point of this essay, including rate (or frequency) coding, temporal coding, and the operation of grid cells.

Rate coding is perhaps the most straightforward way in which neurons represent stimuli. In this scheme, a neuron generates spikes, or action potentials, at a rate that is a monotonic function of the intensity of the stimulus; in other words, the greater the intensity of the stimulus, the more the neuron fires, although “more” may not mean strictly linear. One seminal discovery by Adrian and Zotterman (1926) showed that stretch receptor neurons would increase their firing rate in response to an increase in the mass attached to their innervated muscle. Another example is the discovery that certain neurons in the visual cortex fire most rapidly when presented with visual stimuli rotated at a certain angle (Hubel and Wiesel, 1959). A given neuron will fire most rapidly for stimuli at some fixed angle, and less rapidly as the stimulus angle differs from that “preferred” angle. Thus, firing rate decreases monotonically as stimulus angle decreases from the preferred angle.

Since these early studies, it has become clear that a simple rate code may be too simplistic for many instances of stimulus encoding, so more sophisticated variations of the idea have been proposed; those details do not concern us here. What is important is that, in all of these variations, specific physical properties of neural firing directly represent properties of the relevant stimulus. These are simply analog representations in the sense articulated earlier: the representation (the neural process in question) represents what it does (the stimulus) by monotonically covarying the physical property that does the representing (e.g. the neural firing rate, or the spike density) with the relevant magnitude of the stimulus (muscle stretch, or stimulus orientation). There is no intermediate representation independent of these physical details: the RAL and the IL are the same level.

Temporal coding involves the representation of stimuli via differences in the timing between spikes. In some organisms, a temporal code is used to determine the location of a sound relative to the head. This is represented via the difference in the time it takes for neural spikes generated by one ear to reach a nucleus than for neural spikes generated by the other ear. Roughly speaking, a sound that is closer to say, the left ear, will generate spikes within the left ear sooner than spikes will be generated by the right ear, simply because the right ear is farther from the sound source. Thus, nuclei compute the angle of the sound via the temporal difference between the spikes, relative to the head. These representations are often done via a combination of individual coincidence-detection neurons and their precise physical location relative to one another; again, the details do not matter here, but one example of this kind of representation is found in (van Hemmen, 2006). This representation is a straightforward monotonic one: larger inter-spike differences represent larger differences in arrival times, which then represent larger relative angles of the source sound.

Finally, grid cells are an example of an altogether different—yet still clearly analog—method of neural representation. Neurons in the entorhinal cortex, arranged in a triangular grid-like manner, fire according to where the organism is located in space (Moser et al., 2014). This part of the cortex thus creates a map in a quite literal sense: as the organism moves in a particular direction, neurons will fire in succession along that direction. Figuratively, we can think of the entorhinal cortex as a pixellated

map, where the individual pixels are the firings of individual neurons. Although the neurons themselves do not move, successive firings of neighboring neurons represent movement (much the way that nothing on an LED screen literally moves when a cursor goes from left to right, but movement is suggested by patterns of stationary pixels changing individually). This pattern of firing, due to the structural layout of the entorhinal cortex, represents the organism's movement, direction, and speed, all via the same general analog representational scheme we have encountered above: the physical properties of the neurons and their activity directly represent the relevant features of the stimuli.

These are all examples of types of neural representation that fit into the analog representational and computational scheme, and also demonstrate how this scheme collapses the two Marrian levels we have been discussing. I take this to provide some evidence for the plausibility of the thesis of this article. Next, however, I will turn to somewhat more direct evidence, showing that when neuroscientists explicitly invoke the Marrian framework, they often seem to have in mind exactly the collapse articulated above.

4.2 Neuroscientists on Marr

The first example comes from a recent paper by Barack and Krakauer (2021). This paper argues for the benefit of they call a Hopfieldian approach over a Sherringtonian approach to understanding cognition. While interesting in its own right, of more interest for present purposes is their view of the Marrian levels involved in analyzing each of these approaches. Regarding the Sherringtonian approach, the authors state:

At the implementational level, the Sherringtonian view describes, in biophysical and physiological terms, the neurons and connections that realize a cognitive phenomenon. These descriptions include specific neural transfer functions: the transformations performed by single neurons over their inputs, typically in the dendritic tree or the axonal hillock...At the algorithmic level, the Sherringtonian view appeals to computations performed by networks of nodes with weighted connections between them. In the brain, these nodes are neurons and these connections are synapses. The neurons perform dedicated computational transformations over signals received from other neurons in the network. Explanations of cognition are described in terms of information intake by individual cells, transformation of that information via neural transfer functions and then output to other cells. (Barack and Krakauer, 2021, p. 361).

Note that the description of the implementational and algorithmic levels are nearly identical. This is not to point out any flaws in their characterization: rather, this is a common way of thinking about how Marr's levels apply to neural processes. They make a similar point regarding what they call the Hopfieldian approach:

Implementationally, massed activity of neurons is described by a neural space that has a low-dimensional representational manifold embedded within it. These

neural spaces may be comprised of neural ensembles, brain regions, or distributed representations across the brain. These representations and transformations are realized by the aggregate action of neurons or their subcomponents, but explanations of cognition do not need to include a biophysiological description of neurons or their detailed interconnections...Algorithmically, Hopfieldian computation consists of representational spaces as the basic entity and movement within these spaces or transformations from one space to another as the basic operations. The representations are basins of attraction in a state space implemented by neural entities (be they single neurons, neural populations or other neurophysiological entities) but agnostic to implementational details. (Barack and Krakauer, 2021, p. 363).

Again, both the IL and the RAL level are described as composed of representational spaces, which are abstractions from single or multiple neurons. And again, I take this to be a feature of this kind of investigation, and not a bug. Despite the authors' separation in their description, in both cases the Marrian IL and RAL levels are one and the same.

Another example comes from Grill-Spector and Weiner (2014), in which the authors explicitly adopt the Marrian framework in order to characterize current understanding of the human ventral temporal cortex (VTC). The authors do not describe the RAL and the IL levels in the same terms in the manner of Barack and Krakauer (2021); rather, they characterize the RAL as an *abstraction* of the implementational level. Before getting into the details of what that means, let us see what they say.

In their discussion of the RAL level of the VTC, the authors describe that there are:

Two types of category representations are evident in the VTC: clustered regions that respond more strongly to stimuli of one category as compared to stimuli of other categories (which we refer to as category-selective regions) and distributed representations, that is, neural patterns of responses across the VTC that are common to exemplars of a category. (Grill-Spector and Weiner, 2014, p. 539)

These regions, which have been investigated primarily via fMRI, are then discussed in further detail, which include analyses of their similarities, as well as a discussion of how the stimuli that result in certain patterns of VTC activity can be manipulated (via contrast, or orientation, for example) to result in similar activity patterns.

Interestingly, their discussion of the IL focuses on results about even finer-grained areas of the VTC, again investigated by fMRI. They present evidence of the precise locations in VTC where particular category representation activation occurs. Thus, what (Grill-Spector and Weiner, 2014) characterize as the implementational level is merely a higher-resolution analysis of their RAL level. This is not a bad thing, but it is a thing worth pointing out. The representational properties (i.e. activation of certain regions) are abstractions of the physical properties (i.e. neural firings) of the implementational level. And by “abstraction,” I mean the technical philosophical explication mentioned above: in order to understand certain complex phenomena, it is sometimes necessary to leave out particular details in order to focus on others. But again, abstraction is not implementation.

In both of these instances, the descriptions given by the authors supports the thesis I have argued for here: instead of the RAL and IL being separable, there is a single representational/algorithmic/implementational level. Of course, analyses at that level can happen with higher and lower resolution. However, as articulated in Sect. 2.1, analyses at different resolutions take place *within* a single Marrian level: zooming in at the RAL does not give you an IL analysis, and zooming out does not give you a CL analysis.

5 Conclusion

Analog computation and representation—properly understood—show that Marr’s representational/algorithmic and implementational levels collapse into a single representational/algorithmic/implementational level. Insofar as neural systems are analog systems, Marr’s levels apply to them only in this collapsed form. The view of neural systems as analog is implicit in how neuroscientists understand neural representation, and the view that the representational/algorithmic levels are not distinct from the implementation is implicit in how they appeal to Marr’s levels.

To be clear, I do not claim that every investigative project in neuroscience will neatly fit into the Marrian framework in the first place. Perhaps a level schemata other than Marr’s is better suited for certain research. Furthermore, the entire notion of a “level,” if taken literally, may be misguided: perhaps there are simply *different* explanatory projects that do not have a leveled structure at all (Potochnik and Sanches de Oliveira, 2020). But if the Marrian framework *is* applicable to neural systems, it is only applicable in the collapsed form articulated here.

Although I have focused on Marr’s levels for the purposes of exposition, the collapse involved in analog computational systems generalizes beyond the particularities of the Marrian framework. More generally, when we are clear about the kinds of representations used by neural systems, and clear about the kinds of computations performed on those representation, we may often find that there is no clear distinction between the implementational and representational properties of those systems. This is not to say that it is incorrect or misleading to think of neural systems as representational or computational: we simply need to acknowledge the legitimacy of *analog* representation and computation.

Acknowledgements Thanks to Bryce Huebner, Angela Potochnik, Sarah Robins, and an anonymous reviewer for very helpful comments and feedback on drafts of this paper.

References

- Adams, Z. (2019). The history and philosophical significance of the analog/digital distinction. *Colloquium Series, Pitt Center for Philosophy of Science*
- Adrian, E. D., & Zotterman, Y. (1926). The impulses produced by sensory nerve-endings. *The Journal of Physiology*, 61(2), 151–171. <https://doi.org/10.1113/jphysiol.1926.sp002281>
- Aizawa, K., & Gillett, C. (2009). The (multiple) realization of psychological and other properties in the sciences. *Mind and Language*, 24(2), 181–208. <https://doi.org/10.1111/j.1468-0017.2008.01359.x>

- Barack, D. L., & Krakauer, J. W. (2021). Two views on the cognitive brain. *Nature Reviews Neuroscience*, 22(6), 359–371. <https://doi.org/10.1038/s41583-021-00448-6>
- Bechtel, W., & Shagrir, O. (2015). The non-redundant contributions of Marr's three levels of analysis for explaining information-processing mechanisms. *Topics in Cognitive Science*, 7(2), 312–322. <https://doi.org/10.1111/tops.12141>
- Beck, J. (2018). Analog mental representation. *Wiley Interdisciplinary Reviews: Cognitive Science*, 10(3), e1479–10. <https://doi.org/10.1002/wcs.1479>
- Chrisomalis, S. (2020). *Reckonings: Numerals, cognition, and history*. MIT Press.
- Craver, C. F. (2015). Levels. In T. Metzinger & J. M. Windt (Eds.), *Open MIND*. MIND Group.
- Grill-Spector, K., & Weiner, K. S. (2014). The functional architecture of the ventral temporal cortex and its role in categorization. *Nature Reviews Neuroscience*, 15(8), 536–548. <https://doi.org/10.1038/nrn3747>
- Hubel, D. H., & Wiesel, T. N. (1959). Receptive fields of single neurones in the cat's striate cortex. *The Journal of Physiology*, 148(3), 574–591. <https://doi.org/10.1113/jphysiol.1959.sp006308>
- Isaac, A. (2018). Embodied cognition as analog computation. *Reti, Saperi, Linguaggi*, 7(14), 239–260. <https://doi.org/10.12832/92298>
- Johnstone, M. (2006). Hylomorphism. *Journal of Philosophy*, 103(12), 652–698.
- Lewis, D. K. (1971). Analog and digital. *Noûs*, 5(3), 321–327.
- Love, B. C. (2015). The algorithmic level is the bridge between computation and brain. *Topics in Cognitive Science*, 7(2), 230–242. <https://doi.org/10.1111/tops.12131>
- Lycan, W. G. (1990). The continuity of levels of nature. In *Mind and cognition: A reader* (pp. 77–96). Blackwell
- Maley, C. J. (2011). Analog and digital, continuous and discrete. *Philosophical Studies*, 155(1), 117–131. <https://doi.org/10.1007/s11098-010-9562-8>
- Maley, C. J. (2018). Toward analog neural computation. *Minds and Machines*, 28(1), 77–91.
- Maley, C. J. (forthcoming). Analogue computation and representation. *The British Journal for the Philosophy of Science*. <https://doi.org/10.1086/715031>
- Marr, D. C. (1982). *Vision. A computational investigation into the human representation and processing of visual information*. W. H. Freeman.
- McCulloch, W. S., & Pitts, W. (1943). A logical calculus of ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5, 115–133.
- Moser, E. I., Roudi, Y., Witter, M. P., Kentros, C., Bonhoeffer, T., & Moser, M. B. (2014). Grid cells and cortical representation. *Nature Reviews Neuroscience*, 15(7), 466–481. <https://doi.org/10.1038/nrn3766>
- Peacocke, C. (2019). *The primacy of metaphysics*. Oxford University Press.
- Peterson, G. R. (1967). *Basic analog computation*. The Macmillan Company.
- Piccinini, G. (2020). *Neurocognitive mechanisms: Explaining biological cognition*. Oxford University Press.
- Piccinini, G., & Maley, C. J. (2014). The metaphysics of mind and the multiple sources of multiple realizability. In J. Kallestrup & M. Sprevak (Eds.), *New waves in philosophy of mind*. Palgrave Macmillan.
- Polger, T. W., & Shapiro, L. A. (2016). *The Multiple Realization Book*. Oxford University Press.
- Potochnik, A. (2017). *Idealization and the aims of science*. University of Chicago Press.
- Potochnik, A., & Sanches de Oliveira, G. (2020). Patterns in cognitive phenomena and pluralism of explanatory styles. *Topics in Cognitive Science*, 12, 1306–1320.
- Ritchie, J. B. (2019). The content of Marr's information-processing framework. *Philosophical Psychology*, 76, 1–22. <https://doi.org/10.1080/09515089.2019.1646418>
- Stevens, S. (1946). On the theory of scales of measurement. *Science*, 103(2684), 677–680.
- Thomson, E., & Piccinini, G. (2018). Neural representations observed. *Minds & Machines*, 28(1), 191–235. <https://doi.org/10.1007/s11023-018-9459-4>
- van Hemmen, J. L. (2006). What is a neuronal map, how does it arise, and what is it good for? In J. L. van Hemmen & T. J. Sejnowski (Eds.), *23 Problems in systems neuroscience*. Oxford University Press.
- Von Eckardt, B. (1993). *What Is Cognitive Science?* MIT Press.
- von Neumann, J. (1958). *The Computer and the Brain*. Yale University Press.
- Wolff, J. E. (2020). *The Metaphysics of Quantities*. Oxford University Press.