Cogn Process (2009) 10:53–62
DOI 10.1007/s10339-008-0227-x

RESEARCH REPORT

The role of representation in computation

Gerard O'Brien · Jon Opie

Received: 9 May 2008 / Accepted: 3 September 2008 / Published online: 18 September 2008

Abstract Reformers urge that representation no longer earns its explanatory keep in cognitive science, and that it is time to discard this troublesome concept. In contrast, we hold that without representation cognitive science is utterly bereft of tools for explaining natural intelligence. In order to defend the latter position, we focus on the explanatory role of representation in computation. We examine how the methods of digital and analog computation are used to model a relatively simple target system, and show that representation plays an in-eliminable explanatory role in both cases. We conclude that, to the extent that biologic systems engage in computation, representation is destined to play an explanatory role in cognitive science.

Keywords Analog · Computation · Digital · Representation

Representation in cognitive science

Cognitive Science is a discipline founded on the conviction that natural intelligence can be explained in terms of computational processes that take place in the biologic substrate of cognition. Because computation is governed by the contents of the representations it implicates, or so one influential story goes,[1] the prospects for cognitive science have long been linked to the explanatory credentials of representation. But representation has lately come under attack from several quarters.

Two lines of argument stand out. The first and older stems from the worry that semantic properties are, by nature, incapable of shaping cognitive processes. Indeed, the causal work of a digital computer is normally attributed to the syntactic properties of its internal states, rendering the semantic properties of such states irrelevant to computation. It was precisely this worry that prompted Stephen Stich to advocate the replacement of the computational theory of mind with the syntactic theory of mind (Stich 1983). There have been a number of responses to this way of formulating the problem. One is to insist that representational descriptions of cognitive processes, because they operate at a higher level of generality and present the processes as "rational problem-solving strategies" (Ramsey 1997, p. 41), have an important heuristic value. Another is to appeal to the fact that minds, unlike computer artefacts, causally interact with the environment via sensory and effector systems, and that these hook-ups endow cognitive states with representational content.[2] Neither of these responses is very compelling. The former effectively concedes that representational explanations are inferior to mechanistic or syntactic ones. And with regard to the latter, since mind-world connections are extrinsic to cognitive states, it is hard to see how they can influence the causal powers of such states.[3]

G. O'Brien (✉) · J. Opie
Philosophy, School of Humanities, University of Adelaide, Adelaide, SA 5005, Australia
e-mail: gerard.obrien@adelaide.edu.au
URL: http://arts.adelaide.edu.au/humanities/gobrien/

J. Opie
e-mail: jon.opie@adelaide.edu.au
URL: http://arts.adelaide.edu.au/humanities/jopie/

[1] See, e.g., Cummins and Schwarz (1991, p. 64); Dietrich (1989); Fodor (1975, p. 27); O'Brien and Opie (2006); and Von Eckardt (1993), pp. 97–116.

[2] This is the basis of the "robot reply" to Searle's famous Chinese room argument (Searle 1980).

[3] See, for example, Bickhard (2003); O'Brien and Opie (2004).

A second line of argument derives from the failure of the computational approach to deliver on its promise of explaining intelligence (and the failure of traditional AI to construct deeply intelligent systems). Here, the so-called "knowledge problem" looms large—the problem of equipping a cognitive system with the informational resources to consider and choose appropriate courses of action in real time, in response to open-ended internal goals and subtly changing conditions. According to many cognitive scientists, this problem is so severe as to suggest that cognitive science's flirtation with representation is misconceived: "when we examine very simple intelligence we find that explicit representation and models of the world simply get in the way…Representation is the wrong unit of abstraction in building the bulkiest parts of intelligent systems" (Brooks 1991, p. 140).

The combined effect of these two lines of argument has been the marginalisation of representation in recent cognitive science. Various "anti-representational" approaches, from dynamical systems theory to the embodied-embedded framework, are currently seeking the basis of a new "post-cognitive" paradigm.[4] What unites these approaches is a general skepticism about explaining intelligence in terms of computational processes defined over internal representations.

In our view, without representation cognitive science is utterly bereft of tools for explaining natural intelligence. We would go further: without representation there is no cognitive (as distinct from behavioral, biologic, or just plain physical) science in the first place. It is thus incumbent on those who admire cognitive science to secure the explanatory credentials of representation. The task we have set ourselves in this paper is to take one small step down that road.

Rather than respond to both lines of argument that make trouble for representation, in this paper we aim to show that representation does earn its explanatory keep in computational accounts of cognition. Our strategy will be to examine how the methods of digital and analog computation are employed in modeling a relatively simple target system. In so doing we'll demonstrate that representation is essential to explaining computation in all its forms, but that there are important differences between digital and analog methods. In particular, we'll argue that:

- representation has an indirect role in digital computation (in a sense to be explained), and that this fact motivates the recent skepticism about representation in cognitive science;
- representation has a direct role in analog computation, and hence that an analog conception of cognition promotes a healthy realism about representation.

Despite these differences, we will conclude that to the extent that biologic systems engage in some form of computation, representation is destined to play an explanatory role in cognitive science.

Digital computation

In this section and the next, we examine the characteristics of digital and analog computation by considering how each is used to model the behavior of a simple physical system: a block and spring that together act as a harmonic oscillator. Our analysis permits us to unpack the role of representation in explaining digital and analog modeling, respectively.

The target system

Our target system is a block, of mass m, attached to a vertical spring hanging from an overhead support (Fig. 1). We label the rest position of the block "0" on the vertical axis, and can represent any displacement of the mass as a positive number (when the spring is stretched and the block dips below the horizontal axis) or a negative number (when the spring is compressed and the block rises above the horizontal axis).

Under the right conditions this system, call it T, displays simple periodic behavior. When the block is displaced downwards (say, by gently pulling it with the fingertips) the spring reacts by pulling in the opposite direction, causing the block to move upwards. This motion continues past the rest position so that the spring becomes increasingly compressed, generating a force which sends the block back downwards. The block continues in this manner,

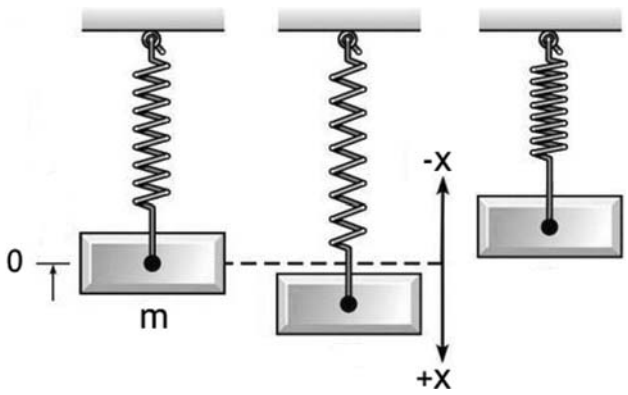

Fig. 1 A block on a spring in three different positions. In the second and third positions the spring is stretched and compressed, respectively, compared to the rest position ($x = 0$)

[4] See, e.g., Beer (1995); Brooks (1991); Clark (1997a, b); Keijzer (2001, 2002); Port and Van Gelder (1995); Van Gelder (1995); Wallace et al. (2007); Wheeler (2005).

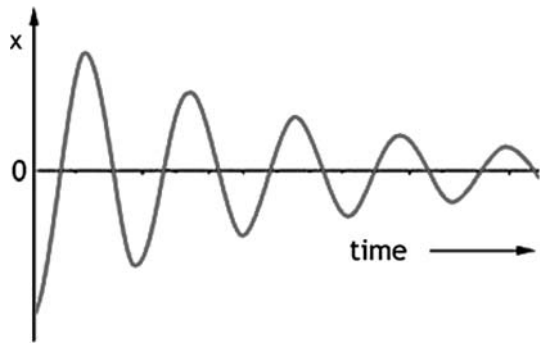

Fig. 2 A graph representing the oscillation of a block on a spring as a function of time

oscillating about its rest position, but eventually comes to a stop as its energy is dissipated, as heat, by friction in the spring (Fig. 2).

Our computational task is to determine the block's position at any time t given its initial displacement and velocity. We have chosen this particular task because there are well-known techniques, both analog and digital, for modeling a harmonic oscillator of this kind. We begin with a digital model.

Digital modeling

The first step in a digital approach to our computational task is to derive a mathematical description of the target system. That work has already been done—the behavior of this system is known to be accurately captured by the following differential equation:

$$m\frac{\mathrm{d}^2x}{\mathrm{d}t^2} + c\frac{\mathrm{d}x}{\mathrm{d}t} + kx = 0. \tag{1}$$

This equation says that the motion of the block depends on its mass, and on two forces: the pull or push generated by the spring as it is alternately stretched and compressed (the third term in the equation), and the internal friction caused by this motion (the second term).[5] The first of these forces is proportional to the distance x of the block from its rest position and the spring constant k, which is a measure of the strength of the spring. The friction is proportional to the block's velocity $\mathrm{d}x/\mathrm{d}t$ and the damping constant c, which is determined by the material constitution of the spring.

To model the behavior of the system using this equation one enters appropriate values of k and c and integrates. There are two ways to do this: using analytic techniques that produce a closed-form solution, namely an equation for the block's displacement expressed as a function of time; or by discrete numerical methods that track the position of the block from moment to moment.[6] The latter involves converting Eq. 1 into finite difference equations in which time is discretized using a fixed minimum interval Δt. Since the block's equation of motion contains a second derivative we end up with a pair of difference equations with which to determine values of x:

$$x_{n+1} = x_n + v_n\Delta t \tag{2a}$$

$$v_{n+1} = v_n\left(1 - \frac{c\Delta t}{m}\right) - x_n\frac{k\Delta t}{m}. \tag{2b}$$

Here, Δt is some small fraction of the block's period of motion and $\mathrm{v} = \mathrm{d}x/\mathrm{d}t$ is its velocity. To calculate the position x_1 and velocity v_1 of the block at time $t = t_0 + \Delta t$, initial values of the position x_0 and velocity v_0 are entered into the right hand side of Eqs. 2a and 2b. The results $[x_1 = x_0 + v_0\Delta t;\ \ v_1 = v_0(1-c\Delta t/m) - x_0k\Delta t/m]$ are fed back into the equations to calculate x_2 and v_2 at $t = t_0 + 2\Delta t$, and so on. In this way, the behavior of the block can be modeled without the use of an analytic solution and, in principle, to any desired accuracy (Fig. 3).

With Eqs. 2a and 2b in hand, there are straightforward pencil and paper techniques for computing the behavior of the block using nothing more than basic arithmetic. Computer science has discovered various ways to mechanize and sequence such operations. Babbage was the first to design a general-purpose device of this kind. Turing later formalized the process by describing an abstract machine capable of automating any sequence of numerical operations that can be finitely specified. What is needed, he realized, is a systematic means of symbolically representing numbers, and a rule-governed mechanism that recognizes and manipulates these symbols purely on the basis of their material properties, but in a fashion that respects their mathematical interpretation.

Physically implementing Turing's machine is no easy feat. First, one must come up with a suitable medium for symbolic representation. One way to achieve this is to partition some continuously variable physical property with a view to provide a semantics for these partitions and their concatenations. In electronic digital computers, for example, the electrical voltages between pairs of wires are partitioned into "high" and "low" voltage states, and symbols take the form of combinations of these two kinds of states. Next, one must find a means of detecting and transforming these symbols on the basis of their macroscopic "syntactic" properties. This is achieved in electronic computers by combining transistors to form "gates" that are differentially responsive to high and low voltages. Finally, one must arrange it so that symbols are transformed in a manner that is sensitive to their intended interpretation. This

[5] For a derivation see e.g. Kibble and Berkshire (2004), Chap. 2. For simplicity we ignore the effect of gravity, which does not alter the sinusoidal character of the block's motion.

[6] Numerical methods are important because for a great many systems there is no analytic solution to the relevant equation(s) of motion.

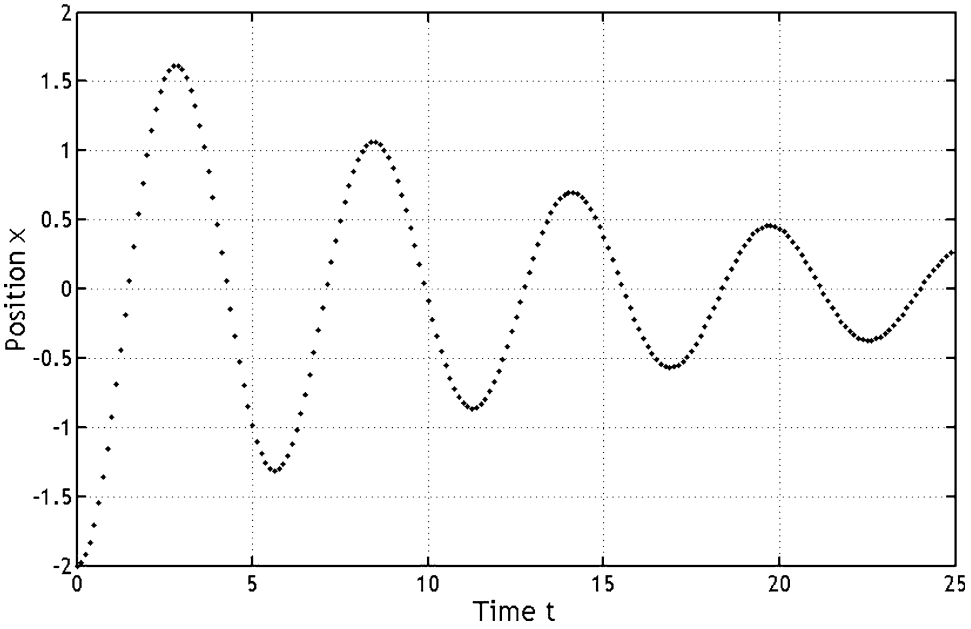

Fig. 3 A plot of the values x_1, x_2, etc. generated by a pair of finite difference equations. The initial position x_0 and velocity v_0 have been set to -2 and 0, respectively. The minimum time interval Δt has been set to about 1/40th of the block's period of oscillation

last step presents a challenge because the syntactic structure of a symbol does not inherently guarantee the semantic coherence of any manipulations that are applied to it. The problem is solved, as it is in the case of pencil and paper calculation, by ensuring that symbol manipulations are governed by a well-chosen system of structure-sensitive rules. One of the great discoveries of computer science is that such rules can themselves be encoded in the form of a special group of symbols known as a "stored program", which is a set of instructions for sequencing more primitive operations (e.g. addition and multiplication). The latter are realized in electronic digital computers as combinations of gates that provide the system with a basic computational competence.

All of these complexities aside, the essential characteristic of the modeling process we have described is that it involves the development of a formal model of the system in whose behavior we are interested. More particularly, to model the block on a spring one must first describe it mathematically and then program the digital computer to "run the maths" (the transformation rules specified in Eqs. 2a and 2b). For this reason, one can think of digital computation as a kind of two-stage representation process: first, one represents the physical system formally, and then one builds[7] a device some of whose states represent the formal (in this case mathematical) entities and whose state-transitions are interpretable as rule-governed transformations of those entities.

[7] In the case of a general purpose digital computer "builds" means programs, but it is in principle possible to construct a special purpose device that is hardwired to perform complex formal operations directly, rather than by sequencing a set of generic primitives.

The role of representation in digital computation

We are now in a position to address the important question: what explanatory role, if any, does representation play in digital computation?

If one focuses on the way that the semantic properties of a symbol are determined, one might conclude that representation has no part to play in explaining digital computation. It is a symbol's intrinsic syntactic properties that determine how it is handled by a digital computer. Such properties provide no guarantee that digital computation will be sensitive to the intended interpretation of symbolic states. Instead, the transformations in a digital computer are governed by carefully chosen rules that dictate how one symbol succeeds another—rules physically realized by, for example, specific combinations of voltage gates wired into the machine. And this implies that it is a physical program and the computational processes it governs that sustains the semantic coherence of a digital computer's state transitions. Computation thus turns out to be explanatorily prior to and independent of representation. To put it more simply, computation sustains representation in digital systems, not the reverse.[8]

[8] If you are not convinced on this point, consider how arithmetic works. The meaning of a numeral such as "21" is not intrinsic to this physical object, but entirely conventional. What guarantees that a calculation taking, say, "21 + 32" as input will produce "53" as output is the structure of the arithmetic rules applied by a competent human computer. These rules are purpose-built to produce numerically coherent results in light of the conventional interpretation of Arabic numerals.

We think this line of reasoning is somewhat misleading; however. The above analysis demonstrates that representation does not play a direct role in digital computation. A digital computer is a simulacrum of a semantic engine, a syntactically driven device masquerading as a respecter of meaning. Nonetheless, we would argue that representation plays a crucial indirect role in digital computation. To see this, consider again the process by which one develops a digital model of some target system.

Constructing a digital model, we argued, is a two-step process: first one formally describes the target system, and then one builds a device that implements this formal model. The reasoning above focuses on the second stage of this process—the implementation of the formal model. And we agree that once a formal model is in place, physically realizing and operating that model is indeed a process in which representation plays no part. But this ignores the first stage of the process—the development of the formal model. It is here that representation has an ineliminable role, because a formal model is a symbolic description of its target, and as such comprises elements that represent features of the target domain. For example, the differential equation (Eq. 1) that describes the behavior of the block, and from which Eqs. 2a and 2b are derived, contains symbols that stand for the mass of the block and its distance from the resting position. Moreover, the shape of a formal model is determined by the representational content of the symbols it contains. Equation 1 has the form it does because it describes the way in which the motion of the block depends on its mass and the two forces acting on it.

Our claim is that representation is essential to formalize a target system, which in turn is essential for constructing a digital model of that system. The role of representation here is indirect in the sense that it constrains the structure of the formal model, and thereby shapes the rules that determine the computer's symbol manipulations, rather than governing those manipulations directly. There is thus a crucial interdependence between representation and computation in digital modeling. Representation plays a role in shaping a formal model of the target system, which when implemented as a program in a digital computer sustains the semantic coherence of (some of) the computer's internal state transitions, thereby licensing us to treat those states as symbolic representations.

It is largely because the explanatory role of representation in digital computation is complex and subtle in this way that skepticism about representation has gained so much traction in contemporary cognitive science. But on the foregoing account, this skepticism is simply not warranted. Without representation, digital computation is formless.

Analog computation

In this section, we explore the role of representation in analog computation by considering an analog model of target system T. We begin by correcting some misconceptions about the nature of analog computation.

Distinguishing between digital and analog computation

The relationship between digital and analog computation is subject to some confusion. It is often claimed that the essential difference between the two styles of computation is that digital computers use discrete variables to represent their targets (e.g. high and low voltage states), whereas analog computers use continuous variables (e.g. voltage, angle, or area). This way of dividing things up is, we think, predicated on the view that a physical system performs a computation just in case its operation can be interpreted as implementing some function. Churchland et al. (1993) express the idea as follows:

> In a most general sense, we can consider a physical system as a computational system just in case there is an appropriate (revealing) mapping between some algorithm and associated physical variables. More exactly, a physical system computes a function f(x) when there is (1) a mapping between the system's physical inputs and x, (2) a mapping between the system's physical outputs and y, such that (3) f(x) = y (1993, p. 48).

This view of computation suggests the following way of distinguishing between digital and analog computers: a physical system is a digital computer if its state variables map onto a discrete function, an analog computer if its state variables map onto a continuous function. Despite the tidiness of this scheme, and a certain degree of acceptance within the computer science community, we believe this way of proceeding is deeply mistaken. It fails on two grounds.

First, there is good reason to reject the idea that computation is merely a matter of implementing a function.[9] Since all law-governed physical systems can be interpreted as implementing some function or other, this view leads to the conclusion that all physical systems are computational. And that conclusion renders the concept of computation explanatorily vacuous. The concept was originally introduced into cognitive science as a way of distinguishing two classes of causal processes: those characteristic of systems

[9] Churchland et al. (1993) do suggest that the mapping between physical variables and algorithm should be "appropriate (revealing)" but these modifiers don't appear to do any real work in their account. Subsequent formulations of the idea, e.g. Chalmers (1994), have not improved the situation.

(such as ovens, cups of tea, and cyclones) that show no signs of intelligence, and those that are the preserve of intelligent systems alone. Computational processes are supposed to be special in some way—in a way, moreover, that provides us with some explanatory purchase with respect to the problem of intelligent behavior. Implementing a function is a ubiquitous feature of nature, so characterizing computation in this way undermines the motivation for introducing the concept in the first place.[10]

Secondly, recourse to the divide between discrete and continuous variables is at odds with the way computer scientists themselves have generally drawn the analog/digital distinction. Digital computers have their origins in the methods of arithmetic. Modern digital machines can be traced back to early devices such as the abacus and the Pascaline[11] that were designed to automate numerical calculation, using machine states (the position of spoked wheels and metal gears in the case of the Pascaline) as symbolic representations of objects and numbers. Analog computation, in contrast, originates with non-symbolic graphical and geometric methods. Consider, for example, the familiar tactic of representing a physical variable as a curve on the plane. If we plot the velocity v of a moving object on one axis and time on the other, it is possible to compute distance travelled by measuring the area under the curve, or acceleration by constructing tangents to the curve (Fig. 4).

These computations employ an analog representing vehicle, a 2–d curve plotted against a pair of linear axes. What makes this vehicle analog is the existence of a relation-preserving mapping between the curve and its target. Velocity is represented as the projection of the curve onto the y-axis, such that relations among velocities correspond to relations among those points. If the velocity at some time t_1 is greater than the velocity at t_2, then its representative point v_1 will be further along the y-axis than v_2; if the velocity at t_3 is mid-way between the first two velocities, then v_3 will lie between v_1 and v_2, and so on. In other words, there is a simple physical analogy between the curve and the variable it represents. Such analogies also underpin the use of a model aeroplane and wind tunnel to predict the effects of air currents on the full-scale system, and the orrery, a clockwork mechanism for representing the relative positions and motions of the planets, to compute changes in planetary positions.

Classic texts in engineering and computer science such as Analog Computation (Jackson 1960), Basics of Analog Computers (Truitt and Rogers 1960) and Analog Methods in Computation and Simulation (Soroka 1954) make precisely this point.

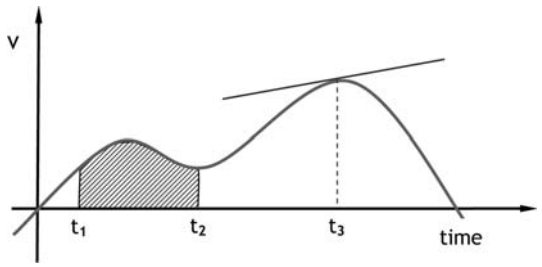

Fig. 4 A graph of velocity versus time. The distance travelled between t_1 and t_2 can be computed by measuring the area of the shaded region. The acceleration at t_3 can be computed by measuring the slope of the tangent to the curve at that point

> Devices that rely… on the analogous relationships that subsist between the physical quantities associated with a computer and the quantities associated with a problem under study are called analog computers (Jackson 1960, p. 1).
>
> All (analog computers) have one characteristic in common—that the components of each computer … are assembled to permit the computer to perform as a model, or in a manner analogous to some other physical system (Truit and Rogers 1960, p. 3).
>
> The term analog means similarity of properties or relations without identity. When analogous systems are found to exist, measurements or other observations made on one of these systems may be used to predict the behavior of the others (Soroka 1954, p. v).

In our terms, what these authors are suggesting is that an analog computer is a device designed to exploit an analogy between the physical properties of a system of representing vehicles and some target system. Digital computers do not work that way. In particular, digital computers deploy symbolic vehicles whose physical properties stand in an arbitrary relationship to the objects they represent.[12] Returning to our original point, the implication here is that the distinction between continuous and discrete variables is not fundamental to the relationship between analog and digital computation. And that is all to the good, because it is not hard to find examples of analog systems that represent their target domain using discrete variables.

Think of the humble clock. Digital clocks, whatever their internal mechanism, represent time using fixed-length numerical symbols. The system of representation here is

[10] For further discussion see O'Brien and Opie (2006). We there suggest an alternative characterization of computation that avoids this criticism.

[11] A mechanical calculator capable of performing addition and subtraction. It was designed by Blaise Pascal to reduce the workload of his father, a tax commissioner.

[12] The relationship is arbitrary in the sense that whether or not one can identify a physical analogy between a system of symbols and their represented objects, no such analogy governs the computational processes defined over those symbols.

discrete in the sense that it has a fixed margin of error: time cannot be represented with accuracy greater than a predetermined smallest interval. Analog clocks do not require symbols, relying instead on some physical process whose unfolding is a monotonic function of time, for example, the changing level of water in a container as it flows through a hole in the base, or the motion of hands rotating about a circular display. Many dial clocks represent the passage of time with smoothly moving hands, and hence continuous variations in angle. However, some have a second or minute hand that moves in discrete steps about the dial. A minute hand that moves in this step-wise fashion is stationary for almost a full 60 s at each mark, then rapidly moves to the next mark where it is again quiescent, and so on. During its rapid sweep from mark to mark the minute hand does of course pass through the intervening positions on the dial, but these do not represent instants of time. In this respect, the system of vehicles here is like that of a digital clock, the one comprising a finite set of positions around a dial, the other, a finite set of symbols, each with a fixed margin of representational error. Nonetheless, all dial clocks are analog, whether discrete or continuous in format, because their meaning is determined by the analogy between the spatial order of positions around a dial and the ordering of moments in time.[13]

Analog modeling

With these preliminaries behind us we return to our computational task: to determine, for any time t, the position of an oscillating block given its initial displacement and velocity. To perform this task using analog means, we need a model system whose behavior is, in appropriate respects, physically analogous to the behavior of the block. A suitable candidate is the simple electric circuit M illustrated below (Fig. 5). M comprises a resistor, a capacitor, an inductor, and a switch. A current I flows through the circuit when the switch is closed. Its behavior is described by the following equation:

$$L\frac{d^2I}{dt^2} + R\frac{dI}{dt} + \frac{1}{C}I = 0. \quad (3)$$

What is immediately apparent, even to those not familiar with differential equations, is that this equation is very similar to Eq. 1, which describes the behavior of the block on a spring. Both have the general form of a damped harmonic oscillator, and the first can be converted into the second by way of the following simple mapping: $x \rightarrow I$, $m \rightarrow L$, $c \rightarrow R$, and $k \rightarrow 1/C$.

These formal similarities reflect the existence of a deep physical analogy between the two systems. A serial capacitor and inductor jointly give rise to oscillatory behavior in a circuit, the capacitor alternately storing and then releasing electrical energy to produce a current, the inductor opposing changes in current and thereby causing it to periodically reverse direction (Fig. 6). The total effect is analogous to the compression and stretching of the spring, and the way this influences the motion of the block. In a circuit, resistance plays a role analogous to friction in that it dissipates electrical energy, converting it into heat. The resistor therefore damps the current in the circuit, eventually reducing it to zero, just as friction eventually brings the moving block to a standstill.

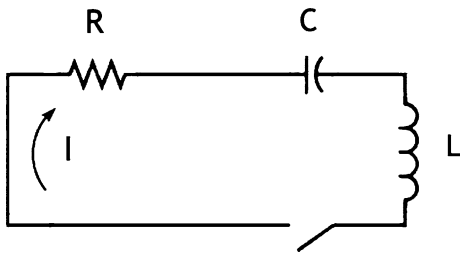

Fig. 5 A circuit M containing a resistor (resistance R), a capacitor (capacitance C) and an inductive coil (inductance L). When the switch is closed a current I flows

As a result of all this, the behavior of M turns out to be ideal for the purpose of representing the motion of the block. With appropriate choices of L, C and R the current in M can be made to behave in such a way that I and x are homomorphic.[14] Formally, this means that there is mapping from the set of values of x into (or onto) the set of values of I that preserves the relationships among the values of x. Informally, it means that if we lay the graph of I (Fig. 6) over the graph of x (Fig. 2) they should line up perfectly wherever they overlap. The upshot is that M can be used to model the target system T, and we can predict or track the behavior of the latter by observing the former.[15]

An objection might be raised at this point concerning the similarity between Eqs. 1 and 3. We argued for the utility of the electrical model of T partly on the grounds of the formal similarity of these equations. Is not it therefore the case that the circuit M, insofar as it plays the role of an analog computer, is simply computing a mathematical function described equally well by Eqs. 1 and 3? This objection misses a crucial and robust distinction between computing an equation and being described by an equation. Again, the classic texts in computer science are clear on this point:

[13] See Lewis (1971) for some further examples of discrete analog representation.

[14] See Bartels (2006) for a nice defense of the view that homomorphism is the basis of representation.

[15] Lest this example seems a bit contrived, it is worth noting that circuits of this kind (albeit somewhat more complex) were used in the 1940s to simulate missile trajectories in order to predict and correct their flight.

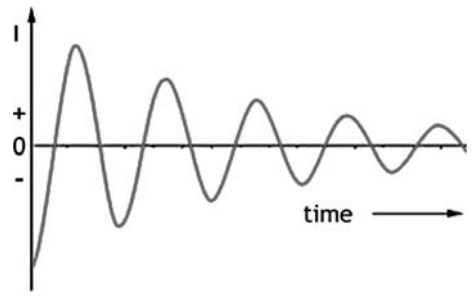

Fig. 6 A graph representing an alternating current *I* in the circuit of Fig. 3. *I* is positive when it flows clockwise around the circuit and negative when it flows counter-clockwise

> … the analog-computer family consists of all those devices in which measurable physical quantities are made to obey mathematical relationships comparable with those existing in a particular problem. … The [general purpose analog] computer consists of a collection of basic building blocks which can be interconnected so that they are governed by the same set of equations as those describing the system to be analyzed (Smith and Wood 1959, p. 1).
>
> Analog models are devices that behave in a fashion analogous to others simply because they obey the same or similar fundamental laws of nature (Truit and Rogers 1960, p. 7).

An analog computer is "governed by" or "obeys" mathematically expressible laws that are similar to those governing the represented system. This does not apply to a digital computer, whose governing equations are, in most cases, those that apply to the components of a digital electronic device. A digital computer, when being used to model some target system, does so by "running the maths", that is, by producing or calculating the values of a mathematical function that describes the target process or object. It does not itself obey those equations, but rather those that govern the dynamics of its representing medium. Putting things the other way around, the circuit M does not calculate the values of its governing equation any more than the planets calculate the solution to Newton's or Einstein's equations of motion in their movements about the sun. A circuit just gets on with behaving like a circuit. When we notice that its behavior is similar to another system of interest we can use it as an analog model. No maths need be involved.

The role of representation in analog computation

With this account of analog computation in the foreground, we can again turn to the critical question: what explanatory role, if any, does representation play in analog computation?

We observed in "Digital computation" that representation has only an indirect role in digital computation. The semantic coherence of a digital computer is not driven by the intrinsic properties of its symbolic representing vehicles, but by a carefully chosen set of program rules. The latter originate with a formal model, a theorists' representation of the target system, which determines the structure of the program and thereby licenses any attribution of content to the digital device. None of this applies to an analog computer. Analog modeling is not a two-stage affair. There is no process of taking a formal description of the target domain and implementing that description as a program. Instead, one simply exploits a physical analogy between the materials at hand and the system of interest. In our example, there is a deep and causally significant analogy between the electric circuit and the block on the spring. And it is this analogy that sustains the semantic coherence of the analog model.

These claims require some unpacking. Physical analogy is a similarity or resemblance relation. Two systems can resemble each other in a variety of ways. They can share first-order properties, such as mass or position, or second-order properties, such as shape or organization. Two balls resemble each other at first-order if they have similar mass, conductivity, or specific heat, or if they share simple relational properties, such as being housed in the same bearing. Second-order resemblance is best characterized as a relation-preserving mapping between two systems. A nice example is the relationship between the thickness of a tree's growth rings and the climatic conditions in which the tree grew. Plentiful seasons produce wide growth rings, whereas drought years produce comparatively narrow rings. The relative thickness of growth rings therefore reflects the variations in the seasons, and permits us to treat the former as a record of the latter.

Second-order physical analogy, which we elsewhere refer to as structural resemblance (O'Brien and Opie 2004), is of particular importance for representation. One system structurally resembles another when physical relations in the first support a relation-preserving mapping between the two. Structural resemblance is the basis of many everyday examples of representation. A road map captures the spatial layout of a streetscape by virtue of the spatial relations among its markings. By preserving relative distance or topology a well-made map permits one to navigate unfamiliar terrain. Likewise, the representing power of a mercury thermometer is due to the way that variations in the height of the column of mercury correspond to variations in ambient temperature. Changes in the level of mercury inform us about temperature changes because of this reliable covariation.

What the foregoing makes clear is that structural resemblance is at the heart of analog representation. In all

of the cases, we have described, from analog clocks to graphs, road maps and mercury thermometers, it is the existence of a relation-preserving mapping between physical properties of the representing vehicles and the modeled system that grounds the content of those vehicles. And the same is true of our principal example. Here, it is the physical relations between the values of the current in the circuit *M* that preserve the relations among the displacements of the bouncing block in *T*, and by virtue of which a particular current carries its representational content. In this respect, analog representation is crucially different from digital representation. Symbols do not carry their meaning in virtue of physical relations to one another, but in virtue of a conventional interpretation which only indirectly governs their role in computation.

Since the content of analog representing vehicles is grounded in structural resemblance, it is determined by intrinsic properties of those vehicles. This fact about structural resemblance has profound consequences for the explanatory role of representation in analog computation.

To begin with, there is a deep physical analogy between *M* and *T*, an analogy formally expressed in Eqs. 1 and 3. The dynamics of the two systems are governed by forces which, although physically distinct (for example, the mechanism in a wire that resists current flow is quite different from the mechanism in a spring that gradually reduces the amplitude of its oscillations) nonetheless interact in similar ways to produce similar behavior in the relevant variables. Structural resemblance is thus at the heart of the generative aspect of computation in an analog computer: the capacity to produce a system of representing vehicles whose physical properties support a desired second order resemblance relation.[16]

Once in place, the representations in a computer typically play some part in producing further representations. But there is a crucial difference between such transformations as they occur in analog and digital computers, respectively. Symbols, like representing vehicles of any stripe, only have causal impacts by virtue of their intrinsic properties. Since the representational content of a symbol is determined by factors extrinsic to that vehicle, its content can have no bearing on what a symbol does. The representing vehicles in an analog computer, on the other hand, acquire content by virtue of their intrinsic physical properties and the resemblance relation(s) these support. Hence, the transformations that occur in an analog computer are driven by the very properties that determine the contents of its representing vehicles. In this sense, representation has a direct role in analog computation.

To make this more concrete, suppose we wish to model the velocity *v* of the oscillating block in *T* using our circuit *M*. The variable *v* is implicit in any analog representation of the block's position, because velocity is a rate of change of position. Consequently, if we add a differentiator to *M*, an electronic device that produces an output proportional to the rate of change of its input, we can harness the current in *M* to produce an analog representation of the block's velocity. What ensures the semantic coherence of this transformation is that current in *M* structurally resembles the position of the block in *T*, and that a differentiator is appropriately sensitive to moment by moment variations in current. Representational content, in the form of current dynamics, is very much in the driver's seat here.

All of this has important implications for cognitive science. We have argued that analog computation is directly shaped by the content determining properties of its vehicles. An analog computer is therefore not a mere semblance of a semantic engine—it is the real thing. Any organism whose inner processes are analog in nature is causally indebted to the semantic properties of its inner states. For this reason, if we adopt the hypothesis that natural intelligence is down to analog processes realized in biologic materials, we thereby commit to a healthy realism about cognitive representation.

Conclusion

We began this paper by describing two lines of argument that have led to the marginalisation of representation in contemporary cognitive science. The first claims that the specifically semantic properties of representing vehicles are irrelevant to computation and hence cannot play an explanatory role in cognitive science. The second holds, more radically, that we should abandon the computational approach to cognition altogether, given the failure of traditional AI to construct deeply intelligent systems. Together these two lines of argument have made popular a number of "anti-representational" approaches in cognitive science which are united by the conviction that cognition cannot be fruitfully understood in terms of computational processes defined over internal representations.

This is a lamentable situation. If it abandons representation, and with it computation, cognitive science will be turning its back on the only good idea we have ever had about how intelligence might arise in a natural system. In this paper, we have begun the task of rescuing cognitive science from this fate. We have shown that semantic properties do earn their explanatory keep in both digital and analog modeling, and hence that representation plays

[16] More generally, the generative aspect of computation is simply the capacity to produce representing vehicles of some kind. This capacity may involve transformations of other representations, but it need not. For example, both analog and digital systems can incorporate transducers, devices that take external signals, say, light waves, and convert them into primary representations.

an ineliminable role in our understanding of computation in both its characteristic forms. However, our response is silent with respect to the second of the arguments mooted above. At this point, therefore, we can only conclude that to the extent that biologic systems engage in computation representation is destined to play an explanatory role in cognitive science. The task of convincing naysayers that computation is indeed the basis of biologic cognition is one that must be left for another time.[17]

References

Bartels A (2006) Defending the structural concept of representation. Theoria 21:7–20

Bickhard MH (2003) Some notes on internal and external relations and representation. Conscious Emot 4:101–110. doi:10.1075/ce.4.1.08bic

Brooks R (1991) Intelligence without representation. Artif Intell 47:139–159. doi:10.1016/0004-3702(91)90053-M

Chalmers DJ (1994) On implementing a computation. Minds Mach 4:391–402. doi:10.1007/BF00974166

Churchland PS, Koch C, Sejnowski T (1993) What is computational neuroscience? In: Schwartz E (ed) Computational neuroscience. MIT Press, Cambridge

Clark A (1997a) The dynamical challenge. Cogn Sci 21:461–481

Clark A (1997b) Being there: putting brain body and world together again. MIT Press, Cambridge

Cummins R, Schwarz G (1991) Connectionism, computation and cognition. In: Horgan T, Tienson J (eds) Connectionism and the philosophy of mind. Kluwer, Dordrecht

Dietrich E (1989) Semantics and the computational paradigm in cognitive psychology. Synthese 79:119–141. doi:10.1007/BF00873258

Fodor JA (1975) The language of thought. Harvester Press, Sussex

Jackson AS (1960) Analog computation. McGraw-Hill, New York

Keijzer F (2001) Representation and behavior. MIT Press, Cambridge

Keijzer F (2002) Representation in dynamical and embodied cognition. Cogn Syst Res 3:275–288. doi:10.1016/S1389-0417(02)00043-8

Kibble TWB, Berkshire FH (2004) Classical mechanics, 5th edn. Imperial College Press, London

Lewis D (1971) Analog and digital. Nous 5:321–327. doi:10.2307/2214671

O'Brien G, Opie J (2004) Notes towards a structuralist theory of mental representation. In: Clapin H, Staines P, Slezak P (eds) Representation in mind: new approaches to mental representation. Elsevier, Amsterdam

O'Brien G, Opie J (2006) How do connectionist networks compute? Cogn Process 7:30–41. doi:10.1007/s10339-005-0017-7

Port R, Van Gelder TJ (1995) Mind as motion: explorations in the dynamics of cognition. MIT Press, Cambridge

Ramsey W (1997) Do connectionist representations earn their explanatory keep? Mind Lang 12:34–66. doi:10.1111/1468-0017.00035

Searle JR (1980) Minds, brains, and programs. Behav Brain Sci 3:417–457

Smith GW, Wood RC (1959) Principles of analog computation. McGraw-Hill, New York

Soroka W (1954) Analog methods in computation and simulation. McGraw-Hill, New York

Stich S (1983) From folk psychology to cognitive science: the case against belief. MIT Press, Cambridge

Truitt TD, Rogers AE (1960) Basics of analog computers. John F. Rider, New York

Van Gelder T (1995) What might cognition be, if not computation? J Philos 92:345–381. doi:10.2307/2941061

Von Eckardt B (1993) What is cognitive science?. MIT Press, Cambridge

Wallace B, Ross A, Davies J, Anderson T (eds) (2007) The mind, the body and the world. Psychology after cognitivism? Imprint Academic, Exeter

Wheeler M (2005) Reconstructing the cognitive world: the next step. MIT Press, Cambridge

[17] We would like to thank two anonymous reviewers of this journal for their helpful comments on an earlier version of this paper.