

A New Problem for Rule Following

Mark Hogarth
Girton College
Cambridge

1. *Introduction*

This is part of an extended argument of mine about the Church-Turing thesis (CTT). In Hogarth 1994 I argued that the thesis is a thoroughly empirical claim. In Hogarth 2004, 2008 I rejected that view, arguing instead that the thesis is really a pseudo-proposition like ‘Australia is below England’, or, better, like ‘Euclidean geometry is the true geometry’. I say ‘better’ because my attitude towards this issue is shaped by an analogy with the concept of geometry. The key idea is that there is no fundamentally privileged computing device (e.g. Turing machine), in just the way that there is no privileged geometry (e.g. Euclidean). From a mathematical viewpoint there are lots of computers, lots of inequivalent ways to execute an algorithm (or procedure or rule). In my previous work I took it that the idea of losing CTT would have important consequences but consequences primarily for computability theory. Here I suggest the consequences extend further and may indeed touch on the nature of pure mathematics itself. Simple counting provides a case study. I also make some remarks about how the analogy with geometry might provide an answer to the stubborn problem of why pure mathematics is applicable to the natural sciences.

2. *The liquefaction of computability*

This line of enquiry began with the discovery of some non-Turing computers within the theory of general relativity (Hogarth 1992, 1994, 2004, 2008; Earman and Norton 1993; Etesi and Nemeti 2001; Nemeti and David 2006). But what I have to say here, and indeed what is important, is not about those computers *per se* but rather about what their existence reveals about the concept of computability (written Computability). In that sense these non-Turing computers are like the first non-Euclidean geometries: what matters, conceptually speaking, is not the real-world accuracy of the models but their existence.

Let us remind ourselves how the concept of geometry came to change. Writing in 1897, when the status of non-Euclidean geometries was still controversial, Bertrand Russell began his book, *An Essay on the Foundations of Geometry*, as follows:

‘When a long established system is attacked, it usually happens that the attack begins at a single point, where the weakness of the doctrine is particularly evident. But criticism once invited, is apt to extend much further than the most daring, at first, would have wished.’

“‘First cut the liquefaction, what comes last,
But Fichte’s clever cut at God himself?’”

‘So it has been with Geometry.’

The liquefaction of Euclidean geometry began at the end of the 18th century, when Gauss questioned one of the axioms of the Euclidean system, the so-called Axiom of Parallels or Euclid’s Postulate 5 (‘Axioms XI’ in some older manuscripts). Gauss

experimented with systems lacking that axiom, but he never published his results for he feared an ‘uproar of the Boeotians’¹. The first publications to present a non-Euclidean system were due Lobachevsky in 1829 and Bolyai in 1832; others followed.

There are two strands of argument in Russell’s book. First, that these new geometries represent real possibilities for the geometry of our world and deciding between them is an empirical matter. Secondly, that any possible geometry must possess constant curvature. This idea Russell held for essentially Kantian reasons. With the arrival in 1915 of Einstein’s general theory of relativity this second strand became untenable, a point Russell himself was quick to acknowledge (Torretti, Chapter 7). Space (and spacetime) could possess variable curvature.

The process of liquefaction therefore advanced further than Russell had anticipated in 1897. After Einstein the term ‘geometry’ became broad and vague. ‘Broad’ because it encompassed a myriad of systems suggested by the sciences; ‘vague’ because the lessons of the past were not to attempt to isolate some ‘essential’ features of geometry (like constant curvature); not, in short, to try to cauterize the concept of geometry. We are now not even tempted to ask what makes a system a ‘geometry’. The term, like the term ‘game’, gets applied because of family resemblances with accepted archetypes. We rightly tend to leave it at that.

One way to phrase the conceptual shift taking us from Euclidean geometry to post-Einstein geometry, is to say that the concept of geometry (written Geometry) became two-sided, with physical geometry on one side and pure geometry on the other. Physical geometry is concerned with modeling the geometry of the physical world; it’s part of physics. Pure geometry is concerned with the mathematical structure of each of the many geometrical models now in the offing; it’s part of pure mathematics.

What I argued in Hogarth 2004, 2008 is that Computability now also looks two-sided, exactly because it has come to look so like Geometry.²

Here I summarise the main points of contact of the two concepts.

¹ Meaning ‘fools’. The ancient Athenians took a dim view of their neighbours in Boeotia.

² This is arguably more than just a metaphor / simile. Computers employ space and time = spacetime = spacetime *geometry*. Thus if Geometry is two-sided, then it is not unreasonable to expect Computability to be two-sided too.

Geometry	Computability
Euclidean	Turing
The various representations of Euclidean geometry by e.g. Euclid, Playfair, Wallis, Saccheri, Riemann.	The various representations of Turing computability by e.g. Church, Kleene, Turing, and Post
‘Euclidean geometry is the true geometry’—call this <i>Euclid’s thesis</i> ; it has a dual role: as a statement of the ‘truth’ of Euclidean geometry, and as an heuristic to complete ‘proofs’	Church-Turing thesis; it has dual role: as a statement of the ‘truth’ of Turing computability, and as an heuristic to complete ‘proofs’.
Geometries of Lobachevsky, Bolyai, Riemann, Einstein, etc.	Quantum computers, relativistic computers (<i>SADs</i>), Davies’s machine
Geometry is two-sided: pure and physical	Computability is two-sided: pure and physical
From pure view point, there is no e.g. Euclidean v. Lobachevsky	From a pure view point, there is no e.g. Turing machine v. <i>SAD</i> ₁
Subjective terms e.g. ‘intuitive’, ‘natural’ fall out of use.	Subjective/vague terms e.g. ‘intuitive’, ‘natural’, ‘mechanical’ fall out of use.
Without Euclid’s thesis, rigorous proofs can be advanced.	Without CTT, rigorous proofs can be advanced.

The ‘*SADs*’ referred to above are relativistic computers that can perform some non-Turing computable tasks. In the Appendix I give a representation of the two simplest, the *SAD*₁ and *SAD*₂, alongside representations of a finite Turing machine (*FTM*) and an ordinary Turing machine (*OTM*). The reader is encouraged to consult the references for further details, but the key point here is that ‘running’ the same algorithm first on the Turing machine and then on the *SAD*₁ will, in general, produce two different results. Hardware matters.

3. Counting

The idea that dispensing with CTT will have implications for only a sliver of mathematics, namely computability theory, is quickly dispelled when one is reminded that pure mathematics is shot through with algorithms (or rules or procedures – I take these to be synonyms).

Take as simple an example as ordinary counting, of the kind a child performs. The procedure (it’s like the Frege successor function) is this:

Begin with $n=1$

Let $n=n+1$

Reveal n

Repeat the second step

Suppose this algorithm is executed on some machine, and an observer consults it from time to time to see what is happening.

One is inclined to say the observer will see something like:

1,2,17,18,101,1201

(Note: the observer observes only from time to time, so some numbers will be missing. And of course the observer does not see the ‘numbers’ as such; rather she must interpret the machine’s output: perhaps 17 dots on a screen is taken to be ‘17’.)

But this entirely depends upon which machine is executing the algorithm. A *FTM* (Appendix) will just stop at 101 (say), so our observer will see:

1,2,17,18,101

An *OTM* (Appendix) will never stop but our mortal observer must, at (say) 1739:

1,2,17,18,101,1739

With a *SAD*₁ our observer sees:

1,2,17,18,101,1201, 1739, ω

This is like the *OTM* case, except now at some point the observer can witness the first ordinal ω (again as naturally interpreted from the machine’s output; see Appendix).

*SAD*₂ goes a step further. Now our observer sees:

1,2,17,18,101,1201, 2015, ω , $\omega+1$, 2ω , ω^2

One tends to think that an algorithm determines the ‘output’. Here we see the output depends, non-trivially, upon the machine too. The output is irreducibly a function of two variables: the algorithm and the machine. An algorithm is by itself indeterminate, and those parts of pure mathematics involving algorithms or procedures or rules really are nothing but squiggles until coupled to a machine. The formalists then were right to take the squiggles in mathematics textbooks to be merely squiggles, but they were wrong in thinking the squiggles are pure mathematics.

This idea undermines a deeply held intuition. We find ourselves saying: but surely the algorithm above just is 1,2,3,...?

All one can say is that executing the algorithm on an *OTM* yields 1,2,3,...

The analogy with geometry may help here. Counting is like building a ladder with numbered rungs. The ladder is built in space (no space, no ladder), and the structure

of the space, together with the building instructions (add another rung), determines the shape of the final ladder.

The fixation with $1,2,3,\dots$ is akin to the fixation with the long Euclidean ladder (in Euclidean space). But – and this is the point – there are other possible ladder structures.

This observation, that algorithm by itself lacks of determinacy, might seem redolent of Wittgenstein's skepticism about rule following:

'... we get [a] pupil to continue a series (say $+ 2$) beyond 1000 — and he writes 1000, 1004, 1008, 1012 (Wittgenstein 1953).'

Wittgenstein is drawing attention to how we use the '+' sign, and consequently what we *mean* by that sign. The pupil's answer, thinks Wittgenstein, does nothing to conflict with past uses (e.g. $10+2=12$) – and so there is nothing 'wrong' with the answer.

But Wittgenstein's problem is of course another problem. For even if the implementation of '+' is unproblematic, transparent, the argument above shows that this mark by itself does no work.

4. How is pure mathematics applicable to the natural sciences?

A stubborn problem in the philosophy of mathematics is this: why is pure mathematics applicable to the natural sciences? Related to this is Benacerraf's problem: how can we come to know the 'inert' objects of pure mathematics?

On the view that CTT holds, or indeed that computation is a transparent process, this is a hard problem. Counting (say) clearly is applicable to the real world and yet is comprised of an algorithm (a pure object) and is driven by a Turing machine (a pure object). How can these two pure objects come together to represent part of the physical world? Put like that, it's a hard problem.

But when Computability is viewed as two-sided, like Geometry, this problem becomes tractable. Counting (say $1,2,3,\dots$) can be treated as a pure system. But that system (algorithm + *OTM*) has, on the other side, a physics, the physics of Turing machines.

This is only a very partial answer to the problem, but it shows where the answer lies (in the physics of machines) – and that was really the problem.

Appendix

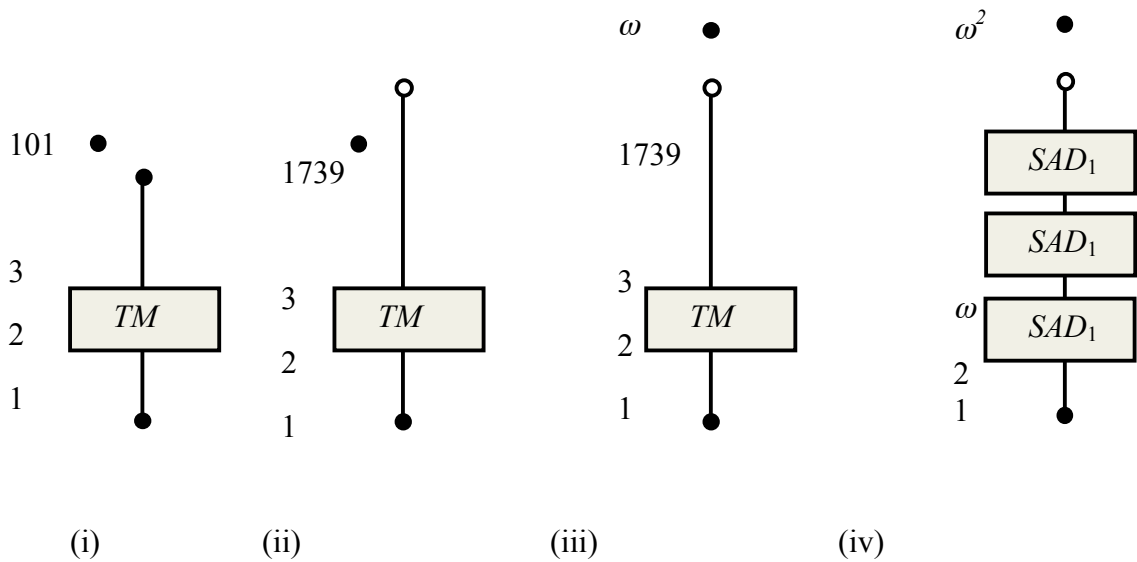


Figure 1. Four different computers, as represented in spacetime.

The figure shows representations of four computing devices. The vertical dimension is time, the horizontal space. A filled dot represents an event; an unfilled dot represents spacetime at ‘infinity’. The unattached filled dot is a typical event on a computer user’s worldline (though the worldline itself is not shown). A line is a worldline of a computer. In (i) the computer stops computing after a finite number of operations. This is the finite Turing machine or *FTM*. In (ii) the computer never stops computing (but the user can access only a finite number of computational steps). This is the ordinary Turing machine or *OTM*. In (iii) the computer is underpinned by a so-called Malament-Hogarth spacetime, which permits the user access to an infinite number of steps. This is called a *SAD₁* computer (because it can decide arbitrary sentences in arithmetic with one quantifier). In (iv) is a *SAD₂*, that is a ‘string’ of *SAD₁*s.

The numbers to the left of each computer are the numbers observed from time to time by a computer user. Of course the ‘numbers’ must be interpreted from the signal data. This holds for 1, 2, etc but also, in (iii), for ω , which is the interpretation of the absence of a signal. Further details can be found in Hogarth (2004).

References

- Davies, E.B. (2001) ‘Building infinite machines’, *British Journal for the Philosophy of Science* 52 671-582.
- Earman, J. and Norton, J., (1993), ‘Forever is a Day: Supertasks in Pitowsky and Malament-Hogarth Spacetimes’, *Philosophy of Science*, 5, 22-42.
- Etesi, G. and Nemeti, I., (2002) ‘Non-Turing computations via Malament-Hogarth space-times’. *International Journal of Theoretical Physics* 41, 2, 341-370.

Hogarth, M., (1994), 'Non-Turing Computers and Non-Turing Computability'. In PSA 1994 vol. 1:126-138. East Lansing: *Philosophy of Science Association*. Ed. D. Hull, M. Forbes, and K. Okruhlik.

Hogarth, M. (2004), 'Deciding Arithmetic Using SAD Computers', *The British Journal for the Philosophy of Science* 55: 681-691.

Hogarth, M., (2008) 'Non-Turing Computers are the New Non-Euclidean Geometries', forthcoming, *International Journal of Unconventional Computing*. forthcoming.

Nemeti, I. and David, Gy., (2006) Relativistic computers and the Turing barrier. *Journal of Applied Mathematics and Computation* 178, 118-142.

Russell, Bertrand., (1996) *An Essay on the Foundations of Geometry*, Routledge.

Torretti, Roberto (1978) *Philosophy of Geometry from Riemann to Poincaré*. Dordrecht: D. Reidel Publishing Co.

Wittgenstein, L., (1953) *Philosophical Investigations*, Blackwell Publishing.