

# Is there any Real Substance to the Claims for a ‘New Computationalism’?

Alberto Hernández-Espinosa<sup>1</sup>, Francisco Hernández-Quiroz<sup>1</sup>,  
and Héctor Zenil<sup>2,3,4</sup>(✉)

<sup>1</sup> Departamento de Matemáticas, Facultad de Ciencias, UNAM, Mexico City, Mexico  
albertoherandezespinosa@gmail.com, fhq@ciencias.unam.mx

<sup>2</sup> Department of Computer Science, University of Oxford, Oxford, UK

<sup>3</sup> Information Dynamics Lab, Karolinska Institutet, Stockholm, Sweden

<sup>4</sup> Algorithmic Nature Group, LABORES, Paris, France  
hector.zenil@algorithmicnaturelab.org

**Abstract.** *Computationalism* is a relatively vague term used to describe attempts to apply Turing’s model of computation to phenomena outside its original purview: in modelling the human mind, in physics, mathematics, etc. Early versions of computationalism faced strong objections from many (and varied) quarters, from philosophers to practitioners of the aforementioned disciplines. Here we will not address the fundamental question of whether computational models are appropriate for describing some or all of the wide range of processes that they have been applied to, but will focus instead on whether ‘renovated’ versions of the *new computationalism* shed any new light on or resolve previous tensions between proponents and skeptics. We find this, however, not to be the case, because the *new computationalism* falls short by using limited versions of “traditional computation”, or proposing computational models that easily fall within the scope of Turing’s original model, or else proffering versions of hypercomputation with its many pitfalls.

**Keywords:** Computationalism · Classical computation · Natural computation · Computability · Turing machine model

## 1 Classical vs. Non-classical Computation

The simplest view of the Turing machine model (TM) construes it as a decision problem solver, tackling such questions as whether a certain string represents a prime number or whether a certain other string belongs to a context-free language. Of course, this view is rather restrictive, as there are many interesting questions that cannot be answered with a simple “yes” or “no”. But TMs can be viewed as mechanisms for calculating functions, with the input string representing the argument(s) of the function and the string left on the tape at halting time

---

Invited contribution to Computability in Europe (CiE) 2017 – ‘Unveiling Dynamics and Complexity’.

representing the result. Given the easy correspondence between natural numbers and finite strings in an alphabet, a TM can be said to calculate a function from natural numbers to natural numbers. Decision problems can be viewed as special cases of functions from natural numbers to natural numbers.

A basic set-theoretical argument tells us that there are many more functions from natural numbers to natural numbers than there are possible TMs and, ergo, that most functions cannot be computed by TMs. The halting problem is one such function.

Within the field of classical computation, and indeed coeval with the introduction of classical computation, certain forms of non-classical computation were devised, such as the oracle machine, which was introduced by Turing himself [31].

Here we do not aim to add to the already lengthy list of possible objections to hypercomputation, which claims the feasibility of computational models that may go beyond the Turing limit in theory but not in practice. Instead we offer an analysis and criticism of supposedly new models of computation that claim to be different from and even to exceed (regardless of whether or not they can be classified as hypercomputation) the classical Turing model in their ability to describe how nature works and—so it is claimed—compute in radical or innovative ways.

## 2 A Brief Roadmap to Computationalism

While there is no current consensus as to the validity of attacks on *classical computationalism*, nowadays many researchers in different fields seem to agree that new models of computation are needed in order to overcome such objections (for a summary of which see [12]).

In this paper we will refer to the former type of computationalism as *classical computationalism* and to the latter type as *new computationalism*. The new computationalist wave is a highly varied mix which encompasses both rejections of Turing’s model and appeals to “natural” computation.

In the decades following Turing’s introduction of his formalization of effective procedure (as defined in [14]) in his seminal paper [30,31], and especially after the widespread and profound success of electronic computers in science and engineering (now universally regarded as incarnations of Turing’s mathematical model), there was a strong impulse to not only use computers in every field within sight but also to view them as models of how things really are, the *computational model of the human mind* being quite probably one of the most, if not the most, iconic instance of this tendency [24]. The process of encoding (or rewriting/or reinterpreting) a problem as a finite sequence of symbols which could be manipulated mechanically by Turing machines in order to solve it is what came to be referred to as *classical computationalism*.

Very soon dissenting voices raised objections based on (controversial) interpretations of Gödel’s theorems [17,23], failures to close the gap between mechanical processing of information and real understanding of it [25,26], and the obvious differences between the way brains process information and the particular

operation of a Turing machine. The crisis in Artificial Intelligence in the 1980s [4] did not help to advance the cause of computationalism, as some early efforts to apply computers to (seemingly) not very complex human abilities like language translation or vision failed miserably. That the objectors to computationalism were not able to present better models of the human mind did not lead them to demur.

The time was ripe for bold proposals to overcome the impasse. Among the most popular were *hypercomputation* and some forms of *natural computing*, together with what we will classify as ‘other models of computation’ based upon variations of the operation of classical models.

### 3 The Uninstantiation of Hypercomputation

A mechanism more powerful than any TM must be able to compute more functions than a TM can. If it merely calculates what a TM does, only (finitely) faster, or more intuitively or with less hassle for its creator, then we cannot say it is computationally more powerful, as it can be simulated by a TM. The Church-Turing thesis states that any formalism capturing what an effective procedure is will be equivalent to a TM [15]. The Church-Turing thesis is much maligned among neo- and hyper-computationalists, but as Sieg [27] has shown (following ideas first advanced by Gandy [9]), it can be reduced to two very basic principles: boundedness and locality conditions. The former implies that a computing device can immediately recognize only a bounded number of configurations, the latter that a computing device can change only immediately recognizable configurations. In (perhaps) oversimplified terms, in order to overcome the TM’s limits, the device must be able to either access an infinite amount of information or must act upon places that are not immediately accessible in a finite number of operations. Sieg’s formulation does not imply a proof of Church’s thesis, but instead establishes a mathematical baseline for the kind of device needed to violate the so-called *Turing limit*.

Such devices may exist or may eventually be created, but they must act very differently from our current computers and cannot be based on trivial variations of classical models.

Some models for hypercomputation include the Oracle Turing Machine model [32], analog recurrent neural networks [29], and analog computation [22].

However, these models are just theoretical constructs, and not only are there no actual devices based on them or physical processes which correspond to them (as far as we know), but there is no prospect of turning them into concrete, viable tools for research in the foreseeable future (even Turing did not have such an eventuality in mind). See Davis’ paper on hypercomputation [5] for a critique of those who think otherwise. As we consider Davis’ analysis quite complete and well founded, we shall dwell no further on this issue and we will conclude this section by saying that non-existence and breach of physical laws (mainly the 2nd. law of thermodynamics) are good reasons to overlook hypercomputation as a meaningful alternative to computation.

More recently, Maldonado [18, 19] has offered a defense of a form of *biological hypercomputation*, claiming that: “[...] life is not a standard Turing Machine”, but rather that living systems hypercompute, and that an understanding of life is reached not by grasping what life is but what it does.

It has even been suggested that phenomena such as death can be sources of a sort of uncomputability, due to the alleged incapability of information theory to describe death or to have it programmed into a system as a desirable property so as to provide meaning to artificial life—just as it does in the case of natural life [8]. Molecular biology, however, can explain death, using straightforward analogies to computation and reprogramming contextualized within information theory [37].

However, that science has not yet fully explained life and death, among other things, does not mean that it will not do so in the future. Thus we consider the claim that neither computationalism nor information theory can explain death (and hence life, according to [8]) shortsighted, if not simply incorrect. Since the discovery of DNA we have known that developmental and molecular biology (and thus biology) are mostly information theoretic, and the more we explore these fields the more we find to confirm our sense that this is indeed the case.

## 4 Natural Computation

Nature is a rich source of ideas and there has lately been a turn toward *natural computation* in the literal sense. Of course, there is nothing wrong with looking to natural phenomena for inspiration. Wolfram takes a very pragmatic approach in his epistemological treatise, a non-classical exploration of the classical computational universe, finding qualitative parallels between nature and computation, with nature harnessing the power of classical computation as a natural source of algorithmic creativity [35].

Others, however, have gone further, offering a divergent notion of computation by attacking classical computation, alleging that a set of constraints that have been in place from the inception of the classical model of computation have handicapped not only the model but the scientific and technological progress of computation as such. The common idea behind most, if not all of these objections to the classical model is that nature does not operate like a Turing machine—because, e.g., nature works in parallel over analog information [7], because nature does more than solve problems [18], and because, it is claimed, there is no way to construct a machine with an infinite tape [7].

Perhaps the most puzzling aspect of the arguments of a group of researchers looking for new notions or types of computation is that while openly accepting that (natural) computing is about information processing (as is classical computation) [6], and also that nature certainly computes (because computers exist in and within nature), they posit a different kind of computation than the classical one [19] while using nature as evidence for non-Turing computation. At the same time none of them specifies exactly what makes this kind of computation distinctive, beyond stressing its difference from a Turing machine (or a trivial

modification of a Turing machine, e.g., a non-terminating one, such as a cellular automaton, which can hardly be classified as non-classical). Such a line of reasoning eventuates in a trivialized natural computation thesis.

Another objection to this view is that by generalizing the notion of computation to any process that transforms its environment, it renders the concept vacuous. The specificity of symbolically encoding problems and solving them by a set of finite, formal rules is then lost. Mechanics, process, transformation *and* computation are all synonyms. Again there is nothing wrong with this per se, but in practical terms this trend does not point out how to attack problems with our current computers, which happen to be (less than ideal) Turing machines. In other words, an extension of the concept of computation should require an enhancement of computers. At present it is highly debatable where this enhancement will come from, but this line of thought definitely takes us back to hypercomputational ground.

## 5 Not More but Equally Powerful

Finally there is an abundance of computational models that have sometimes been touted as more powerful alternatives to Turing machines, but on inspection turn out to be mathematically equivalent to Turing's model. Of course, a different model may give us a new, powerful insight into an aspect of computation obscured by the rigidity of Turing machines (mind you, they were supposed to be rigid in the extreme). This is the case with numerous models of concurrent and distributed computation [21]. But this does not mean that these models solve problems a Turing machine cannot. Being the good theoreticians they are, the people behind concurrent or distributed models have not made any such claim.

Of course, the mathematical equivalence between the TM model and many others (programming languages, concurrent computation, etc.) does not mean that the latter are superfluous. They were introduced to solve real, important problems for which TMs did not provide a clear or manageable way of expressing the actual questions. If, for instance, you are trying to capture the fundamental issues of communication and synchronization dealt with by the  $\pi$ -calculus [20] you will not get very far by encoding them as a string to be manipulated by a sequential TM with its mind-boggling (and mostly irrelevant) details, even if this were theoretically possible. In other words, we are not challenging the utility of alternatives to the TM, only the claim that some models can do what no TM can *in principle*.

Among other models is the *Interaction machine* model that “extend[s] the Turing Machine model by allowing *interaction*, i.e. input and output actions (read and write statements) determined by the environment at each step of the computation” [33,34],  $\pi$ -calculus, “a mathematical model of processes whose interconnections change as they interact” [20,21]. Scott Aaronson offers a whimsical characterization of the Interactive model [1]. It is puzzling that interactive computations cannot simply be viewed as independent classical computations.

Moreover, software such as operating systems are implementations of highly interactive programs. They were introduced early in the development of the first computers, and concurrent computation is an active area of research where these kinds of questions are addressed within a very classical—so to speak—framework. Nothing in Turing’s model prevents an external observer or machine from interacting with the working tape of the original machine, thus effectively interacting with the machine itself.

Another model is the *Inductive computing* model in the context of what the author has called ‘super-recursive algorithms’ [2]. In many respects, the inductive machine model is not comparable to classical computation, but there is one respect in which it is not that far removed from a certain form of such computation. Inductive computation does not produce a definitive output and is thus similar to transducers, and like cellular automata they do not terminate, suggesting a transducer or cellular automaton-type of computation that supposedly generalizes the classical Turing machine model.

Many of the objections based upon models such as that of Gurevich [10], even when deployed with intent to disprove the Church-Turing Thesis [11]—in what constitutes a clear misunderstanding of the philosophical basis and content of said thesis [28]—are based upon, for instance, the argument that Turing machines cannot deal with structures other than strings on tapes, even when trivial modifications that preserve all of their classical properties have been made, modifications that in no way imply the invalidity of the original TM model [16] (for example, models of Turing machines operating on grids preserving, say, algorithmic information properties [36]). This does not mean, however, that such models cannot be useful, a case in point being high-level descriptions of classical computation, with Gurevich’s [10] model being put to very practical use nowadays in software engineering, as a tool for software modelling.

## 6 Old Dogs, New Tricks

A standard for surpassing the Turing model and disproving the Church-Turing thesis must entail something far more stringent than trivial modifications of classical computation. Of course, the main question is what constitutes a non-trivial modification of the classical model, a modification that does more than simply introduce an infinite element which merely takes the purportedly feasible new model into the *hypercomputation* category.

We consider the use of the expression “more powerful” as merely metaphorical unless specifics are provided as regards what makes a given model more powerful. Likewise, if as soon as such a model is instantiated it merely becomes as powerful as or else not comparable to classical computation.

An example of a trivial modification (to modern eyes) that has been accepted as not leading to more computational power other than speed-up is the use of additional machine tapes.

In the same fashion then, when it is claimed that concurrent computation is more powerful than TMs as TMs are crippled by their ‘sequentiality’ (bearing in mind that though concurrency can be properly simulated in sequential

TMs, doing so is very cumbersome), we do not consider such an objection to be an objection in principle, as it is not related to the inability of the model to undergo minor changes without changing anything more than the details of its operation. Or that object oriented programming is more powerful because it is heuristically superior to clumsy TMs (although again TMs can simulate object oriented programming—with overhead). Similarly, objections concerning speed, illustrated by, e.g., quantum computing, do not fall into the category of fundamental challenges to the model, having to do only with its operation.

A model claimed to be more powerful than classical models is the so-called ‘actor’ model that, according to its originator, was inspired by physics, including general relativity and quantum mechanics. It was also influenced by the programming languages Lisp, Simula and early versions of Smalltalk, as well as capability-based systems and packet switching. Its development was “motivated by the prospect of highly parallel computing machines consisting of dozens, hundreds, or even thousands of independent microprocessors, each with its own local memory and communications processor, communicating via a high-performance communications network.” [3]

According to Hewitt [13], “concurrency extends computation beyond the conceptual framework of Church, Gandy, Gödel, Herbrand, Kleene, Post, Rosser, Sieg, Turing, etc. because there are effective computations that cannot be performed by Turing Machines. . . . [and where] computation is conceived as distributed in space where computational devices communicate asynchronously and the entire computation is not in any well-defined state. (An *actor* can have stable information about what it was like when it received a message.) Turing’s Model is a special case of the Actor Model.” [13]

It appears trivial to most computer scientists that these models can be simulated by classical models (e.g. by dovetailing on parallel computations on different inputs stored in different tapes) as long as there are not an infinite number of interactions or an infinite number of actors acting at the same time that would violate the boundedness and locality principles of feasible models [27, 28].

## 7 Conclusion

This paper does not attempt to disprove the existence of ways of overcoming the limitations of the traditional Turing machine model or to provide a survey of models of computation purported to go beyond the Turing model (whether claiming the status of hypercomputation or not). Instead, it attempts to be a reminder of what those limitations are and how far some claims have gone in trying to establish a new type of computationalism, claims that are often, if not always, (mistakenly) predicated on the apparent weakness of classical models, weaknesses that are in fact only weaknesses of orthodox interpretations of their operating details.

There are very good theoretical models of what life looks like that purport to surpass the Turing machine model, but we are still far from being able to put any of these models into practice, assuming it will ever be possible to do so.

Every few years we see a claim of this sort, and its technical merits should be assessed in order to (most improbably) accept it or (as has been usual hitherto) debunk it.

Clearly we have not gone in for clever new theorems or innovations attempting to analyze specific proposals that have been floated, which you may find disappointing (just good old theory). For our part we are even more disappointed at not being able to acknowledge the appearance of novel and more solid ideas and have not felt compelled to spend time producing a theorem to show that a classical Turing machine can, for example, simply be extended to operate on grids and other structures and still preserve its classical nature by virtue of preserving all the theory of computation derived for it, respecting hierarchies and at most achieving speed-up gains.

The common denominator of all these attacks on classical computation, including Church's thesis, is the impression they create of refuting an opponent's argument though the arguments refuted are not ones that have actually been advanced by anyone—what is called a *straw man fallacy*. In effect disputes are generated where there are none. For example, no serious researcher has ever suggested that the mind, nature or the universe operates or is a mechanical incarnation of a (universal) Turing machine.

In order to build sound objections against classical computation and computationalism, we conclude that it is thus necessary to represent it in its full spectrum, and not to adopt an old, abstract, symbol-manipulation view of computation that is out of date or else has been oversimplified for other purposes.

## References

1. Aaronson, S.: The Toaster-Enhanced Turing Machine, Blog entry. <http://www.scottaaronson.com/blog/?p=1121>. Accessed 11 Feb 2017
2. Burgin, M.: Super-Recursive Algorithms. Monographs in Computer Science. Springer, New York (2005)
3. Clinger, W.D.: Foundations of actor semantics. AITR-633 (1981)
4. Crevier, D.: AI: The Tumultuous History of the Search for Artificial Intelligence. Basic Books, New York (1993)
5. Davis, M.: The myth of hypercomputation. In: Teuscher, C. (ed.) Alan Turing: Life and Legacy of a Great Thinker, pp. 195–211. Springer, Heidelberg (2004)
6. Dodig-Crnkovic, G.: Significance of models of computation, from turing model to natural computation. *Minds Mach.* **21**(2), 301–322 (2011)
7. Fresco, N.: Physical Computation and Cognitive Science. Studies in Applied Philosophy, Epistemology and Rational Ethics, vol. 12. Springer, Heidelberg (2014)
8. Froese, T.: Life is precious because it is precarious: Individuality, mortality, and the problem of meaning. In: Dodig-Crnkovic, G., Giovagnoli, R. (eds.) Representation and Reality: Humans, Animals and Machines, Springer (in press)
9. Gandy, R.: Church's thesis and the principles for mechanisms. In: Barwise, H.J., Keisler, H.J., Kunen, K. (eds.) The Kleene Symposium, pp. 123–148. North-Holland Publishing Company (1980)
10. Gurevich, Y.: Sequential abstract state machines capture sequential algorithms. *ACM Trans. Comput. Log.* **1**(1), 77–111 (2000)



11. Gurevich, Y., Dershowitz, N.: A natural axiomatization of computability and proof of the church's thesis. *Bull. Symbolic Logic*. **14**(3), 299–350 (2008)
12. Hernández-Espinosa, A., Hernández-Quiroz, F.: Does the principle of computational equivalence overcome the objections against computationalism? In: Dodig-Crnkovic, G., Giovagnoli, R. (eds.) *Computing Nature Turing Centenary Perspective. Studies in Applied Philosophy, Epistemology and Rational Ethics*, pp. 225–233. Springer, Heidelberg (2013)
13. Hewitt, C.: What is computation? Actor model versus turing's model. In: *A Computable Universe, Understanding Computation and Exploring Nature as Computation*. World Scientific Publishing Company/Imperial College Press, Singapore (2013)
14. Hilbert, D.: Neubegründung der Mathematik: Erste Mitteilung. *Abhandlungen aus dem Seminar der Hamburgischen Universität* **1**, 157–177 (1922)
15. Kleene, S.C.: *Introduction to Metamathematics*. North-Holland, Amsterdam (1952)
16. Langton, C.G.: Studying artificial life with cellular automata. *Physica D: Nonlinear Phenom.* **22**(1–3), 120–149 (1986)
17. Lucas, J.R.: Minds, machines and Gödel. *Philosophy* **36**(137), 112–127 (1961)
18. Maldonado, C.E., Gómez-Cruz, N.A.: Biological hypercomputation: A concept is introduced (2012). arXiv preprint [arXiv:1210.4819](https://arxiv.org/abs/1210.4819)
19. Maldonado, C.E., Gómez-Cruz, N.A.: Biological hypercomputation: A new research problem in complexity theory. *Complexity* **20**(4), 8–18 (2015)
20. Milner, R.: *Communicating, Mobile Systems: The Pi Calculus*. Cambridge University Press, Cambridge (1999)
21. Milner, R., Parrow, J., Walker, D.: A calculus of mobile processes. *Inf. Comput.* **100**(1), 1–40 (1992)
22. Mycka, J., Costa, J.F.: A new conceptual framework for analog computation. *Theoret. Comput. Sci.* **374**(1–3), 277–290 (2007)
23. Penrose, R., Mermin, D.: The emperor's new mind: Concerning computers, minds, and the laws of physics. *Am. J. Phys.* **58**(12), 1214–1216 (1990)
24. Putnam, H.: *Representation and Reality*. A Bradford Book, Cambridge (1988)
25. Searle, J.R.: Minds, brains, and programs. *Behav. Brain Sci.* **3**(3), 417–424 (1980)
26. Searle, J.R.: Is the brain a digital computer? *Proc. Addresses Am. Philos. Assoc.* **64**(3), 21–37 (1990). American Philosophical Association
27. Sieg, W.: Church without dogma: Axioms for computability. In: Cooper, S.B., Löwe, B., Sorbi, A. (eds.) *New Computational Paradigms*, pp. 139–152. Springer, New York (2008)
28. Sieg, W.: Axioms for computability: Do they allow a proof of church's thesis? In: Zenil, H. (ed.) *A Computable Universe: Understanding and Exploring Nature as Computation*. World Scientific Publishing Press, Singapore (2013)
29. Siegelmann, H.T.: Recurrent neural networks and finite automata. *Comput. Intell.* **12**(4), 567–574 (1996)
30. Turing, A.M.: On Computable Numbers, with an Application to the Entscheidungsproblem. *Proc. London Math. Soc.* **2**(42), 230–265 (1936)
31. Turing, A.M.: On Computable Numbers, with an application to the Entscheidungsproblem: A correction. *Proc. London Math. Soc.* vol. 2 (published 1937). **43**(6), 544–546 (1938)
32. Turing, A.M.: Systems of logic based on ordinals. *Proc. London Math. Soc.* **2**(1), 161–228 (1939)
33. Wegner, P.: Why interaction is more powerful than algorithms. *Commun. ACM* **40**(5), 80–91 (1997)

34. Wegner, P.: Interactive foundations of computing. *Theoret. Comput. Sci.* **192**(2), 315–351 (1998)
35. Wolfram, S.: *A New Kind of Science*. Wolfram Media, Champaign (2002)
36. Zenil, H., Soler-Toscano, F., Delahaye, J.P., Gauvrit, N.: Two-dimensional kolmogorov complexity and validation of the coding theorem method by compressibility. *Peer J. Comput. Sci.* **1**, 23 (2015)
37. Zenil, H., Schmidt, A., Tegnér, J.: Causality, information and biological computation. In: Walker, S.I., Davies, P.C.W., Ellis, G. (eds.) *Information and Causality: From Matter to Life*. Cambridge University Press (in press)

Unveiling Dynamics and Complexity

13th Conference on Computability in Europe, CiE 2017,

Turku, Finland, June 12-16, 2017, Proceedings

Kari, J.; Manea, F.; Petre, I. (Eds.)

2017, XIII, 401 p. 57 illus., Softcover

ISBN: 978-3-319-58740-0