Five views of hypercomputation

BRUNO LOFF[1,2]*, JOSÉ FÉLIX COSTA[1,2]†

[1] *Instituto Superior Técnico, Dept. of Mathematics, Technical University of Lisbon*
[2] *CMAF, Complexo Interdisciplinar, Universidade de Lisboa*
Lisboa, Portugal

We overview different approaches to the study of hypercomputation and other investigations on the plausibility of the physical Church–Turing thesis. We *propose* five thesis to classify investigation in this area.

Halloween.
Sly does it. Tiptoe catspaws. Slide and creep.
But why? What for? How? Who? When! Where did it all begin?
"You don't know, do you?" asks Carapace Clavicle Moundshroud climbing out of the pile of leaves under the Halloween Tree. "You don't REALLY know!"

— *Ray Bradbury*, The Halloween Tree

* email: bruno.loff@gmail.com
† email: fgc@math.ist.utl.pt

1 INTRODUCTION

In (5; 6) Martin Davis criticises the relevance of so-called hypercomputation. Is hypercomputation a new theory to understand the mathematics of hyper-degrees? Is it a theory about concrete computation in the physical world? We review the hypercomputation concept to realise that five main views, or trends, can be found in the literature.

People outside the field mainly think that hypercomputation is related to the real numbers, and that models are endowed with hypercomputation by the fact that they incorporate real parameters, using the infinite amount of information contained in some incompressible real number.

This view is wrong! The hypercomputational contents of some computational models have nothing to do with the infinite amount of information processed by an algebra of real numbers. Martin Davis' paper (5) has *been abused* by others, both in oral presentation and paper citation. The relevance of real numbers is low, and the hypercomputational contents of these models arise from treating the reals as an oracle, or advice: *it is only possible to use a finite part of the oracle (or real number) during a finite computation*.

In these pages we overview some models of hypercomputation in order to remove misunderstandings in research. We will organise our discussion into five sections, each of which describes one main trend in the field.

2 HYPERCOMPUTATION AS UNSIMULABLE PHENOMENA

Our first view of hypercomputation emerges from the detailed study of physical theories with the aim of finding whether computers can or cannot simulate what these theories intend to model. It is based on the following:

> **Thesis S (for 'simulation').** There are non-simulable phenomena in physical theories. Thus (maybe) physical reality has hypercomputational content.

The work we will review is grounded on Turing's definition of computable real numbers and Lacombe–Grzegorczyk's characterisation of computable real functions, and it is worthwhile to recall the following definitions:

Definition 1 *(I) A sequence of rational numbers,* $(q_n)_{n\in\mathbb{N}}$*, is called* **computable** *if for some* 1*-ary total recursive functions* a, b, c*:*

$$q_n = (-1)^{a(n)}\frac{b(n)}{c(n)}.$$

(II) $r \in \mathbb{R}$ *is a* **computable real number** *if some computable sequence of rational numbers,* $(q_n)_{n\in\mathbb{N}}$*, is such that, for all* $n \in \mathbb{N}$*,* $|q_n - r| \leqslant 2^{-n}$*.*

(III) $X \in \mathbb{R}^k$ *is a* **computable tuple of real numbers** *if every element in the tuple is a computable real number.*

(IV) $f : \mathbb{R}^k \to \mathbb{R}$ *is a* **computable function** *if there are three recursive functionals* $A, B, C : \mathbb{R}^k \times \mathbb{N} \to \mathbb{N}$ *such that, for every tuple* $\vec{x} \in \mathbb{R}^k$*,*

$$\left|(-1)^{A(\vec{x};n)}\frac{B(\vec{x};n)}{C(\vec{x};n)} - f(\vec{x})\right| \leqslant 2^{-n}, \text{ for every } n \in \mathbb{N}.$$

Having established this notion of computable real function we can investigate whether the evolution of a system specified with computable input and obeying certain physical laws may be uncomputable. In (23) Marian Pour-el and Ian Richards describe a 2-ary computable real function g such that none of the (non-unique) solutions of the differential equation

$$h(x) = 0 \qquad \partial_x h(x) = g(x, h(x))$$

is computable. This result is not tied to any physical theory. In a latter article (24) the same authors describe a certain 3-ary computable function f such that the three-dimensional wave equation

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} - \frac{\partial^2 u}{\partial t^2} = 0$$

with the initial conditions

$$u(x, y, z, 0) = f(x, y, z) \qquad \frac{\partial u}{\partial t}(x, y, z, 0) = 0$$

gives a unique solution which is not computable. In fact, $u(0, 0, 0, 1)$ is a non-computable real number. This result is extended in (25) to show that given a compact set D, f can be constructed so u is not computable in any neighbourhood of any point in D. The uncomputability of the solutions of the wave equation has been thoroughly investigated by Weihrauch and Zhong in (31). They study computability of functions in different topological spaces to conclude that if $f \in C^k(\mathbb{R}^3)^\star$ is computable and all its kth order partial derivatives are also computable, then the solution of the wave equation above must be of class $C^{k-1}(\mathbb{R}^4)$ and all its partial derivatives of order up to $k - 1$

* $C^k(\mathbb{R}^3)$ denotes the space of continuously differentiable functions over $\mathbb{R}^3$ of degree k.

must also be computable. We can conclude — as Pour-el and Richards already had — that the partial derivatives of f are not computable, and this is, in fact, where the uncomputability of u comes from. In regard to the physical feasibility of a *wave computer*, Weihrauch and Zhong write:

> *In summary, even under very idealising assumptions about measurements and wave propagation in reality, it seems to be very unlikely that the Pour-el and Richards counterexample can be used to build a physical machine with a 'wave subroutine' computing a function which is not Turing computable. We may still believe that the* [physical] *Church–Turing Thesis holds.*

However, Pour-el and Richard's constructions show that it is not difficult to incorporate non-computable phenomena in physical theory. Penrose (21) also comments Pour-el and Richards' results:

> *[...] their 'peculiar' kind of initial data is not 'smoothly varying' (that is, not twice differentiable), in a way that one would normally require for a physically sensible field.*

Another investigator of Thesis S is Warren D. Smith. In his paper (28) Smith describes a Newtonian system of N point-masses in 2-dimensional Euclidean space which cannot be simulated by a Turing machine. We fix the masses of the system to certain rational numbers, allowing such a system to be specified by a tuple of $4N$ real numbers — two real numbers for the position vector and two for the velocity vector, times N particles: we call such a tuple a *specification*. The system follows the law of motion,

$$\ddot{\vec{x}}_i = G \sum_{j \neq i} m_j \frac{\vec{x}_j - \vec{x}_i}{\|\vec{x}_j - \vec{x}_i\|^3}.$$

Smith's results tell us that there is no general algorithm which is, given a computable specification, able to decide the question: *Does any body in the system intersect the unit ball in the first second of the system's evolution?* [†] So being able to specify an N-body Newtonian system in 2-dimensional Euclidean space to an arbitrarily high precision is not enough to be able to predict what the system's behaviour will be. On (28), Smith writes:

[†] We can assume that if the body does intersect the unit ball in the system then it will also intersect $B(0, 1-\epsilon)$ for some predetermined $0 < \epsilon < 1$, i.e., Smith's undecidability result does not stem from the fact that the body may intersect the unit ball arbitrarily near the perimeter.

The present paper demonstrates (I claim) that unsimulable physical systems exist in Newton's laws of gravity and motion for point masses. However, it does not appear to demonstrate, that, if we lived in a universe governed by those laws, we could actually build a device with super-Turing computational power.

In the next section we will study what such devices may look like. Following his exploration of the uncomputability of Newtonian systems, Smith reformulates the laws of movement to include relativistic phenomena, showing that under these laws computable input will result in computable output. In section 6 we will give a brief account on the idea that non-computability should be an essential part of any 'ultimate' physical theory.

3 HYPERCOMPUTATION AS COMPUTATION WITHOUT A PROGRAM

The second view we present pertains to the idea that hypercomputation can be performed by abstract physical devices, i.e., by machines with a behaviour obeying certain laws of physics. It is nevertheless impossible to program these machines, as their specification cannot be described finitely:

Thesis N (for 'not programmable'). Hypercomputation is about computations which can not be folded into a program, but can be performed by an ABSTRACT physical machine.

The theory of analogue recurrent neural networks (ARNNs) was developed by Hava Siegelmann and Eduardo Sontag (see (27)). Each ARNN consists of a finite number of *inputs* and *units* which evolve in discrete time steps. In a network with N units and M inputs the *state* of each unit is updated at every step, following the rule

$$x_i(0) = 0 \quad x_i(t+1) = \sigma\left(\sum_{j=1}^{N} a_{ij}x_j(t) + \sum_{k=1}^{M} b_{ik}u_k(t) + c_i\right),$$

where $x_i(t)$ is the state of the ith unit at time t, $u_k(t)$ is the value of the kth binary input at time t and a_{ij}, b_{ik}, c_i are real constants called *weights*. Above, σ is the *activation function*, given by

$$\sigma(x) = \begin{cases} 0 & \text{if } x \leqslant 0, \\ x & \text{if } 0 < x < 1, \\ 1 & \text{if } x \geqslant 1. \end{cases}$$

ARNNs provide a uniform model for expressing varied computational power. The following table shows the set of languages decidable by an ARNN under different restrictions. It is possible to restrict the set of weights — imposing, for instance, that $a_{ij}, b_{ik}, c_i \in \mathbb{N}$ — or the number of steps the network is allowed to take. Bellow, t will be a time constructible function.

Set of weights	Time restriction	Computational power	
$\mathbb{N}$	none	Regular languages	
$\mathbb{Q}$	none	Recursive languages	
$\mathbb{Q}$	t	$\mathrm{DTIME}(t)$	(A)
$\mathbb{R}$	polynomial	$\mathrm{P/poly}$	(B)
$\mathbb{R}$	none	All languages	(C)

FIGURE 1
Computational power of ARNNs under various restrictions.

Up to this point there is no claim for hypercomputation. The original work of Siegelmann and Sontag is, first and foremost, a study in structural complexity, with relevant results for non-uniform complexity classes: there is, we insist, no mention of hypercomputation. Nevertheless, the model has endured severe criticism (5), mostly targeting Hava Siegelmann's 1995 article entitled *Computation beyond the Turing limit* (26). In this article, and in the expanded version found in (27), Siegelmann shows that ARNNs are computationally equivalent to *analogue shift maps*, a model of computation which extends Cris Moore's *generalised shift maps* (20). Both of these systems can be implemented by an abstract optical device based on parabolic mirrors: the device is set up according to the weights of the network. We discuss the following

> **Common misconception.** The device cannot be built because it is impossible to adjust the mirrors to the required infinite precision (e.g. 8, p. 100).

To understand why this is not the true reason we must explain (roughly) a few details of how we can prove (B). Although not simple to prove, it is straightforward to show schematically that ARNNs with real weights can decide $\mathrm{P/poly}$ in polynomial time. From (A) above we can see that there must be a neural net which can decide the circuit value problem in polynomial time, call it $\mathbf{NCVP}$. We also know that $\mathrm{P/poly}$ is the class of languages decidable

by polynomial size circuits. The following proposition, which we can derive from the work by Siegelmann and Sontag, provides the final ingredient:

Theorem 2 *(1) There is a bijective encoding from the set of families of circuits to the* 9*-Cantor subset of* $[0, 1]$*;*

(2) if α *encodes a family* $(A_k)_{k\in\mathbb{N}}$ *of polynomial size circuits, then the code of the* n*th circuit can be found among the first* $p(n)$ *algarisms of the decimal expansion of* α*, where* p *is a polynomial depending on* $(A_k)_{k\in\mathbb{N}}$*, and furthermore*

(3) we can construct an ARNN — call it $\mathbf{N}\alpha$ *— with weights in* $\mathbb{Q} \cup \{\alpha\}$ *to extract, given input* n*, the code of* A_n *in a number of steps bounded by* $p(n)$*.*

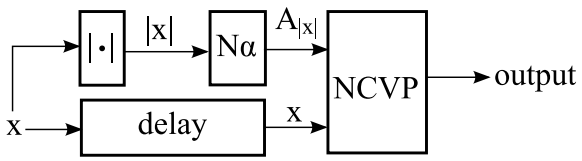

FIGURE 2
P/poly with an ARNN.

So given a set in $A \in \mathrm{P}/\mathrm{poly}$, we find the real number α that codes for the family of polynomial size circuits which decide A, build the net $\mathbf{N}\alpha$ and the net $\mathbf{NCVP}$. Then we build an ARNN which, given input x, feeds $|x|$ into $\mathbf{N}\alpha$, and when $\mathbf{N}\alpha$ has extracted $A_{|x|}$, feeds $\langle A_{|x|}, x\rangle$ into $\mathbf{NCVP}$, from where it obtains the output. This is schematised in Fig. 2.[‡]

The common misconception mentioned above can be dispelled by the following theorem, which tells us that linear precision is sufficient to simulate an ARNN.

Theorem 3 *The output of an ARNN after* t *steps is only influenced by the first* $\mathrm{O}(t)$ *algarisms in the decimal expansions of the weights.*

This means that, given input x, we can simulate a polynomial number $p(|x|)$ of steps of an ARNN in polynomial time, given as advice $\mathrm{O}(p(|x|))$ bits

[‡] While the figure is not completely faithful to Siegelmann and Sontag's work — there is no *delay unit* in the ARNN they present — we decided to favour clarity over accuracy.

specifying the weights of the network: we can simulate ARNNs in P/poly. ☞ So in order to make the optical device work correctly on inputs of size n it would only be necessary to adjust the mirrors of the device up to a polynomial precision in n.

☞ While this may still be to much precision to allow for implementation, *the true reason this device cannot be built in order to do hypercomputation is more fundamental: the algarisms of the weights themselves cannot be known in advance*. Before even considering the technical problem of building the machine with the mirrors precisely adjusted, we stumble upon the evident difficulty of finding out the angle of adjustment itself.

This clears the possibility of building such a machine for any useful purpose. Nevertheless we can do the following philosophical thought experiment: imagining that such a machine is build, with the mirrors positioned more-or-less randomly, is it performing hypercomputation in some sense?

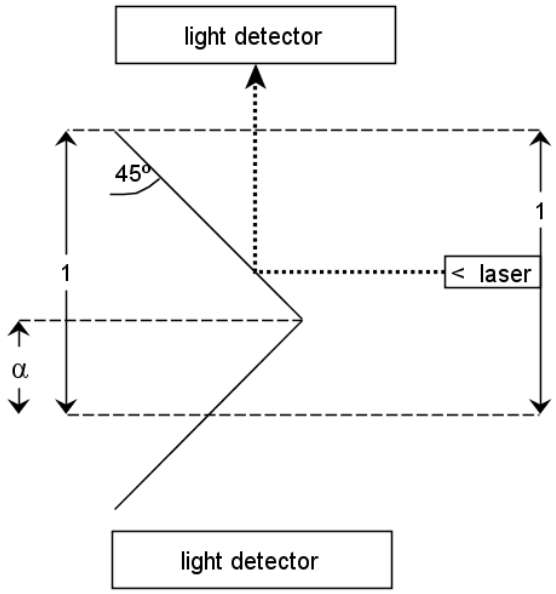

FIGURE 3
A schematic drawing of the scatter machine.

Another model which follows Thesis N is the scatter machine, recently introduced by Edwin Beggs and John Tucker (1). The scatter machine con-

sists of two straight mirrors forming a wedge, a laser and two light detectors laid out as in Fig. 3.¶ The laser can be placed (for instance) in any position along the line having a power of 2 as denominator. In this case it is possible to obtain the value of α (in the figure) with an exponentially small error in polynomial time: we begin with an interval $[a, b] = [0, 1]$, fire the laser to a, b and $\frac{a+b}{2}$ and decide in the following step to change one of a or b to $\frac{a+b}{2}$, based on which detectors reported a hit in each of the three firings. Should we couple this mechanical device to a Turing machine able to move the laser in this way and read from the detectors, we can in polynomial time decide exactly P/poly.

4 HYPERCOMPUTERS

The most controversial view of hypercomputation pertains to the following:

> **Thesis P (for 'programmable').** Hypercomputations can be specified for an apparatus which is not physically unplausible.

There are two main paths leading to this thesis: Tien Kieu's adiabatic quantum computer (16; 17) and Turing machines in curved space-time (13; 7; 32). Several objections have been put forward to Kieu's approach (30; 12; 29; 9), some have been answered (15; 19; 18), and the discussion is still active. Problems of a physical nature — related to blue shift — have been put forward against the use of curved space-time, which the authors believe to have solved.

5 HYPERCOMPUTATION AS AN ORACLE OR ADVICE

The following thesis is also extremely debatable.

> **Thesis O (for 'oracle').** The Universe has non-computable information content (which may be used as an oracle to build a hypercomputer).

Cooper and Odifreddi (3), for instance, have suggested similarities between the structure of the Universe and the structure of the Turing universe.

¶ In (1) the scatter machine is described as a Newtonian system: point particles replace the laser light and perfectly elastic barriers replace the mirrors. For this discussion the slight differences are irrelevant: we nevertheless mention that some issues which arise by the possibility of the point mass hitting the vertex of the barrier are solved by using mirrors and lasers.

Calude (2) investigates to what extent quantum randomness can be considered algorithmically random. The search for a physical oracle was proposed by Jack Copeland and Dianne Proudfoot (4). Their article and subsequent work have been severely criticised (5; 11) for historical and technical errors. There is, however, an appealing aesthetical side to what Copeland and Proudfoot proposed. The oracles of Turing machines are often seen and taught as abstract theoretical entities, merely technical devices. We nevertheless feel that it may be an interesting and beautiful endeavour to regard oracles as a natural phenomena and study the oracles that arise in nature. We will give the example of Stonehenge. The archaeological monument 'Stonehenge', located near Amesbury in the English county of Wiltshire, was built in three main phases from 3000BC to 1500BC. In the first main stage, called 'Stonehenge I', the monument was composed of around eighty standing stones. Fifty six of these stones where laid out in a circle around the centre of Stonehenge I (see Fig. 5), and are called Aubrey holes. We can number the Aubrey holes and make use of three tokens, placing each token in one of the holes: if the ith token is in the nth Aubrey hole, we say that *the ith register holds the number n*. In this peculiar way, Stonehenge I can be seen as a resource bounded implementation of a (Turing-universal) 3-counter machine. It is known, after the work of Gerald Hawkins, Fred Hoyle and others (10; 14), that it is possible to use Stonehenge as a predictor of lunar and solar eclipses. From the point of view of the Earth both the Moon and the Sun follow approximately elliptical orbits, as shown in Fig. 4, which cross at the nodes N and N'. Suppose the moon is passing through N. Then a solar eclipse will occur if the sun is no further than $15^\circ$of N, and a lunar eclipse happens if the sun is within $10^\circ$of N'. If the moon is passing through N' the situation is reversed. One can then wait for a solar eclipse, set the three tokens in the appropriate Aubrey hole, and use the following:

Hoyle's algorithm

1. The first token, a little stone for instance, is moved along the Aubrey holes to keep track of the 28 day lunar cycle. We move the first token two places every day, since $56/2 = 28$.

2. The second token counts the days of the year. Since $56 \times 13/2 = 364$, we move the second token two places every thirteen days.

3. The third token will represent one of the nodes, say N. N and N' themselves rotate around the Earth, describing a full cycle (called a Metonic

cycle) every 18.61 years. So we will move the third token three times every year, because $56/3 = 18.67$.

4. Eclipses occur when the three tokens become aligned with each other up to one Aubrey hole to the right or to the left.

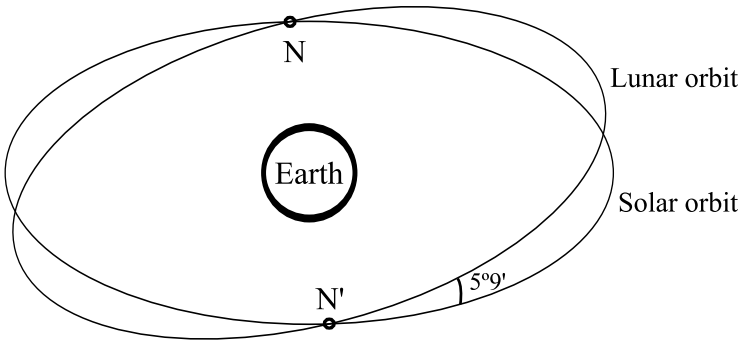

FIGURE 4
The approximate orbits of the Moon and the Sun around the Earth.

Ignoring the error for now, we conclude that simple modulo 56 arithmetic is enough to predict every eclipse with one single necessary *input*: the day of a solar eclipse when one sets the tokens in the first Aubrey hole. Now we introduce the *Oracle*: to the Northeast of Stonehenge I there is a 5 meter tall stone, called the 'Heelstone'. In the morning of the Summer solstice the sun (our oracle) is born slightly to the north of the Heelstone. To know the exact day of the Summer solstice we wait for the day when the sun rises behind the Heelstone. The sunrise should then proceed north for a few days, and then back south. We count the number of days between the first sunrise behind the Heelstone and the second. The day of the summer solstice happened in the middle of these two events. [§] With this information we can calibrate the sun token to enough precision every year, so that Stonehenge I can predict eclipses indefinitely. [||]

[§] The image in the first page illustrates where the sun would be, next to the heelstone, in the day of the summer solstice.

[||] The calibration procedure explained in (14) is slightly more complicated and detailed: we only illustrate it here. The remaining tokens can also be calibrated using other oracles: the phases of the moon give the adjustment the first token and the precise day in which a solar eclipse occurs allows for calibration of the third token.

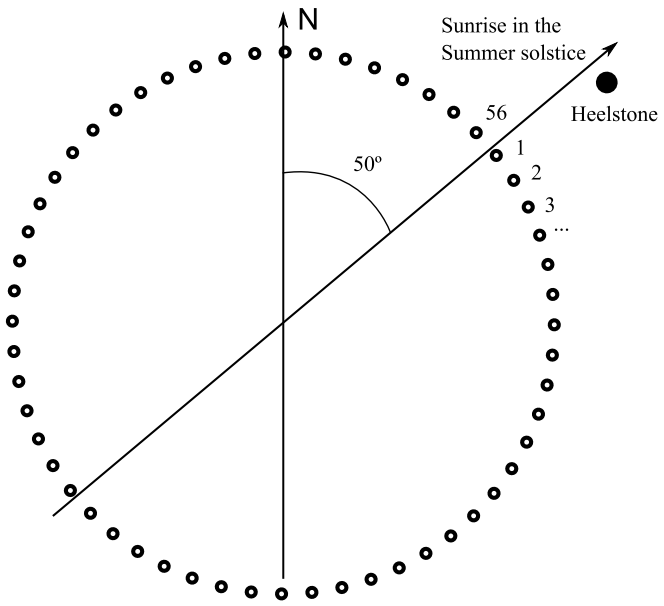

FIGURE 5
A schematic drawing of Stonehenge I.

We have described an unusual form of computation, aided by an an unusual oracle. In our discussion we could have replaced Stonehenge I with a modern computer, and our oracle could be, for the sake of an example, a link with a satellite telescope. While it seems natural to consider the Sun as an oracle in the Stonehenge I algorithm described above, calling 'oracle' to this satellite link can feel awkward — one may prefer to call it 'input'. We defend that apart from their symbolic value, these two sources of information have the same nature. It is custom to consider that input is finite, and given prior to the computation, but the sunrises or the satellite link give — in principle — an unbounded amount of data. They act as an oracle. Without these oracles both Stonehenge I and our modern computer would eventually be incapable of predicting eclipses, although the modern computer could remain providing accurate predictions for hundreds of years.

Consider the physical Church–Turing type of thesis, opposite to Thesis S: *the physical world is simulable.* This thesis leads us to conclude that one could, in principle, construct a Turing machine that could successfully pre-

dict eclipses forever, without the use of any oracle.[#] Being able to predict eclipses indefinitely, however, would not imply that the physical world is simulable, unless the prediction of planet alignments (called *conjunctions*) is, in some sense, *completeł* for the simulation problem.

6 HYPERCOMPUTATION AS A THEORY OF EVERYTHING

Roger Penrose (21; 22) has put forward several arguments against the idea of a computable Universe. He distinguishes that uncomputability should manifest itself in *essential* and *non-essential* ways. Pour-el and Richard's uncomputable solution of the wave equation is given as an example of the later. Deeply rooted in Penrose's work is a belief that the essential uncomputability of the Universe will provide evidence for:

> **Thesis E (for 'everything').** The final theory of physics is to be found uncomputable.

The uncomputability described in this theory would presumably aid in explaining the — seemingly non-algorithmical — phenomenon of conscious intelligence, which places Roger Penrose in the debate against the strong A.I. thesis.

7 ACKNOWLEDGEMENTS

Bruno Loff would like to thank CMAF for the office space and the other fabulous resources made available to him. The authors are grateful to Mike Stannett for having invited them to submit this paper after the *International Interdisciplinary Workshop on Future Trends in Hypercomputation*, Sheffield, September 2006. José Félix Costa is also very indebted to the *United Grand Lodge of England* for providing him with the Stonehenge symbol. In this sense we thank to two great men of England, the archaeologist Sir Gerald Hawkins and the (deceased) astronomer Sir Fred Hoyle.

REFERENCES

[1] Edwin Beggs and John Tucker. (September 2006). Experimental computation of real numbers by newtonian machines. Technical report, University of Wales Swansea.

[#] We feel it is justified, in this discussion, to abstract from the cosmological knowledge that eclipses will not happen forever.

[2] Cristian S. Calude. (2004). Algorithmic randomness, quantum physics, and incompleteness. In M. Margenstern, editor, *Machines, Computations and Universality (MCU'2004)*, volume 3354 of *Lecture Notes in Computer Science*, pages 1–17.

[3] S. Barry Cooper and Piergiorgio Odifreddi. (2003). *Computability and Models, Perspectives East and West*, chapter Incomputability in Nature, pages 137–160. University series in mathematics. Springer.

[4] Jack Copeland and Diane Proudfoot. (April 1999). Alan Turing's forgotten ideas in computer science. *Scientific American*, 280:99–103.

[5] Martin Davis. (2006). The myth of hypercomputation. In Christof Teuscher, editor, *Alan Turing: the life and legacy of a great thinker*, pages 195–212. Springer.

[6] Martin Davis. (July 2006). Why there is no such discipline as hypercomputation. *Applied Mathematics and Computation*, 178(1):4–7.

[7] Gábor Etesi and István Németi. (February 2002). Non-Turing computations via Malament–Hogarth space-times. *International Journal of Theoretical Physics*, 41(2):341–370.

[8] Antony Galton. (July 2006). The Church–Turing thesis: still valid after all these years? *Applied Mathematics and Computation*, 178(1):93–102.

[9] Amit Hagar and Alexandre Korolev. (February 2006). Quantum hypercomputability? *Minds and Machines*, 16(1):87–93.

[10] Gerald Hawkins. (1965). *Stonehenge Decoded*. Doubleday.

[11] Andrew Hodges. The professors and the brainstorms. Online at `http://www.turing.org.uk/philosophy/sciam.html`.

[12] Andrew Hodges, (December 2005). Can quantum computing solve classically unsolvable problems? Archive preprint `http://arxiv.org/quant-ph/0512248`.

[13] Mark Hogarth. (1994). Non-Turing computers and non-Turing computability. *Proceedings of the Biennial Meeting of the Philosophy of Science Association*, 1:126–138.

[14] Fred Hoyle. (1972). *From Stonehenge to modern cosmology*. W. H. Freeman.

[15] Tien Kieu, (November 2001). Reply to "the quantum algorithm of Kieu does not solve the Hilbert's tenth problem". Archive preprint `http://arxiv.org/abs/quant-ph/0111020`.

[16] Tien Kieu. (November 2002). Quantum hypercomputation. *Minds and Machines*, 12(4):541–561.

[17] Tien Kieu. (2003). Quantum algorithms for the Hilbert's tenth problem. *International Journal of Theoretical Physics*, 42:1461–1478.

[18] Tien Kieu, (February 2006). On the identification of the ground state based on occupation probabilities: an investigation of Smith's apparent counterexamples. Archive preprint `http://arxiv.org/abs/quant-ph/0602145`.

[19] Tien Kieu, (February 2006). Reply to Andrew Hodges. Archive preprint `http://arxiv.org/abs/quant-ph/0602214`.

[20] Cris Moore. (May 1990). Unpredictability and undecidability in dynamical systems. *Physical Review Letters*, 64(20):2354–2357.

[21] Roger Penrose. (1989). *The emperor's new mind*. Oxford University Press.

[22] Roger Penrose. (1989). *Shadows of the mind*. Oxford University Press.

[23] Marian Pour-El and Ian Richards. (1979). A computable ordinary differential equation with possesses no computable solution. *Annals of Mathematical Logic*, 17:61–90.

[24] Marian Pour-El and Ian Richards. (1981). The wave equation with computable initial data such that its unique solution is not computable. *Advances in Mathematics*, 39(4):215–239.

[25] Marian Pour-El and Ning Zhong. (1997). The wave equation with computable initial data whose unique solution is nowhere computable. *Math. Log. Q.*, 43(4):499–509.

[26] Hava Siegelmann. (April 1995). Computation beyond the Turing limit. *Science*, pages 545–548.

[27] Hava Siegelmann. (1999). *Neural networks and analog computation: beyond the Turing limit*. Birkhäuser, Cambridge, MA, USA.

[28] W. D. Smith. (July 2006). Church's thesis meets the N-body problem. *Applied Mathematics and Computation*, 178(1):154–183.

[29] W. D. Smith. (July 2006). Three counterexamples refuting Kieu's plan for "quantum adiabatic hypercomputation"; and some uncomputable quantum mechanical tasks. *Applied Mathematics and Computation*, 178(1):184–193.

[30] Boris Tsirelson, (November 2001). The quantum algorithm of Kieu does not solve the Hilbert's tenth problem. Archive preprint `http://arxiv.org/abs/quant-ph/0111009`.

[31] Klaus Weihrauch and Ning Zhong. (2002). Is wave propagation computable or can wave computers beat the Turing machine? *Proceedings of the London Mathematical Society*, 85(2):312–332.

[32] Philip Welch, (2006). The extent of computation in Malament-Hogarth spacetimes. Technical report.