Algorithmic Social Sciences Research Unit

ASSRU

Department of Economics
University of Trento
Via Inama 5
381 22 Trento Italy

DISCUSSION PAPER SERIES

10 – 10

NOTES ON THE MATHEMATICAL FOUNDATIONS OF ANALOGUE COMPUTATION♣

K. Vela Velupillai

DECEMBER 2010

♣ The issues discussed in this paper are based on work that has been central to the Computable Economics research project initiated by Stefano Zambelli and myself over a quarter of a century ago. Two of the central mathematical concepts that have informed the theoretical framework of Computable Economics are the Church-Turing Thesis and Universal Computation. The contents of this paper are a first foray into an investigation into the `computability' properties of the Phillips Machine, considered as a special kind of General Purpose Analog Computer (GPAC). That I am deeply indebted to Stefano Zambelli, for invaluable intellectual companionship, for over a period of over twenty five years, goes without saying. The problem, however, is that I am unable to blame him for the obviously remaining infelicities in the paper -- nor can I blame our competent graduate students, Kao Selda and V. Ragupathy for those very infelicities.

Abstract

Digital computing has its mathematical foundations in (classical) recursion theory and constructive mathematics. The implicit, working, assumption of those who practice the noble art of analog computing may well be that the mathematical foundations of their subject is as sound as the foundations of the real analysis. That, in turn, implies a reliance on the soundness of set theory plus the axiom of choice. This is, surely, seriously disturbing from a computation point of view. Therefore, in this paper, I seek to locate a foundation for analog computing in exhibiting some tentative dualities with results that are analogous to those that are standard in computability theory. The main question, from the point of view of economics, is whether the Phillips Machine, as an analog computer, has universal computing properties. The conjectured answer is in the negative.

1 Introductory Notes

> "More specifically, do computer trajectories 'correspond' to actual trajectories of the system under study? The answer is sometimes no. In other words, there is no guarantee that there exists a true trajectory that stays near a given computer-generated numerical trajectory. ....
>
> Therefore, the use of an ODE solver on a finite-precision computer to approximate a trajectory of a .... dynamical system leads to a fundamental paradox. .... Under what conditions will the computed trajectory be close to a true trajectory of the model?"
>
> [14], p.961.

There are two caveats, from an analog computing point of view, that should be remembered: one, the 'computer trajectories', referred to above, are those generated by a digital computer; secondly, the 'fundamental paradox' should not be relevant for an analog computer (subject, of course, to machine precision, which is different from finite-precision in a digital computer).

But there could be other paradoxes between the mathematical theory of an analog computing machine and its theoretical limits and its actual trajectories. The aim of this paper is to pave an introductory path towards a discussion of this possible paradox, in the context of the actual functioning philosophy and epistemology of a Phillips Machine.

There is a clear acknowledgement – albeit somewhat belated – to the long and rich tradition of considering computability and complexity of models defined over $\Re$, in a variety of recursive and computable analytic frameworks, by Smale, a leading advocate of side-stepping the Turing model of computation for computation over the reals, in [16], p.61:

> "Indeed, some work has *now been done* to adapt the Turing machine framework to deal with real numbers........ Thus, the foundations are probably being laid for a theory of computation over the real numbers."

This acknowledgement comes in 1991 after almost six decades of work by recursive and computable analysts to *adapt* the Turing model to domains over $\Re$. Weihrauch, himself a notable contributor to this adapting tradition refutes what I can only call a 'preposterous' claim in [1], p.23 (italics added):

> "A major obstacle to reconciling scientific computation and computer science is the present view of the machine, that is *the digital computer*. As long as the computer is seen simply as a *finite* or *discrete* object, it will be difficult to systematize numerical analysis. We believe that the Turing machine as a foundation for real number algorithms can only obscure concepts."

I can only endorse, wholeheartedly, Weihrauch's entirely justifiable claim that the theory presented in his book, [18], p.268, 'refutes [the above] statement.'

But even the gods 'nod', sometimes! Systematizing numerical analysis is one thing; computing over the reals may well be quite another thing – especially since the definition of real numbers can be approached from a variety of mathematical and logical points of view. If we concentrate on computing over the reals, then the long and noble analog computing tradition, using only a finite number of discrete objects, can accomplish – at least theoretically – anything that can be achieved by taking the numerical analysis route.

One of the recurrent themes in the work by Smale and his co-workers on *Real Number Computation* ([1]) is the need for a model of computation over the reals so that the classic problems of mathematical physics, or applied mathematics, like those posed by the need to solve, numerically, ordinary differential equations, defined over $\mathbb{R}$, can be achieved in a theoretically satisfactory way. Consider the following system of nonlinear ordinary differential equations, the so-called Rössler System:

$$\frac{dx}{dt} = -(y + z)$$

$$\frac{dy}{dt} = x + 0.2y$$

$$\frac{dz}{dt} = 0.2 + z(x - 5.7)$$

Suppose a *General Purpose Analogue Computer* (*GPAS*)[1] is defined in terms of the usual *adders*, *multipliers* and *integrators* as the elementary units, analogous to the elementary operations defining a Turing Machine or a partial recursive function ($\mu-$recursion, minimalization, etc.,). Then it can be shown that a *GPAS* consisting of 3 adders, 8 multipliers and 3 integrators (the symbolic definitions are given in the next section) can simulate the above Rössler System (see Figure 1). The intermediate step of having to use a numerical algorithm to implement a computation, as in a Turing Machine model, is circumvented in the analog computation model.

In implementing a particular computation in the Phillips Machine, there is no need for the intermediate step of writing an algorithm for the numerical computation of the dynamics of the differential equation. The differential equation, in turn has the dual purpose of being a model of the economy and that of the dynamics of the machine. In the case of the Phillips Machine a linear differential equation model plays this dual role. However, despite the ingenuity with which Phillips constructed his machine, so that the dynamics of the flows and the stocks can be approximated by linear differential equations, the actual functioning of the Phillips Machine remains stubbornly nonlinear.

Is it not better, then, to avoid the linearizations and try to model, directly and faithfully, the nonlinear dynamics of the machine? If this is done, and with the added advantage of circumventing the intermediate step of implementing a numerical algorithm, would not the model dynamics, as represented, in real time, by the analog machine – in our case, the Phillips Machine – depict the

[1] See the formal definition in the next section.

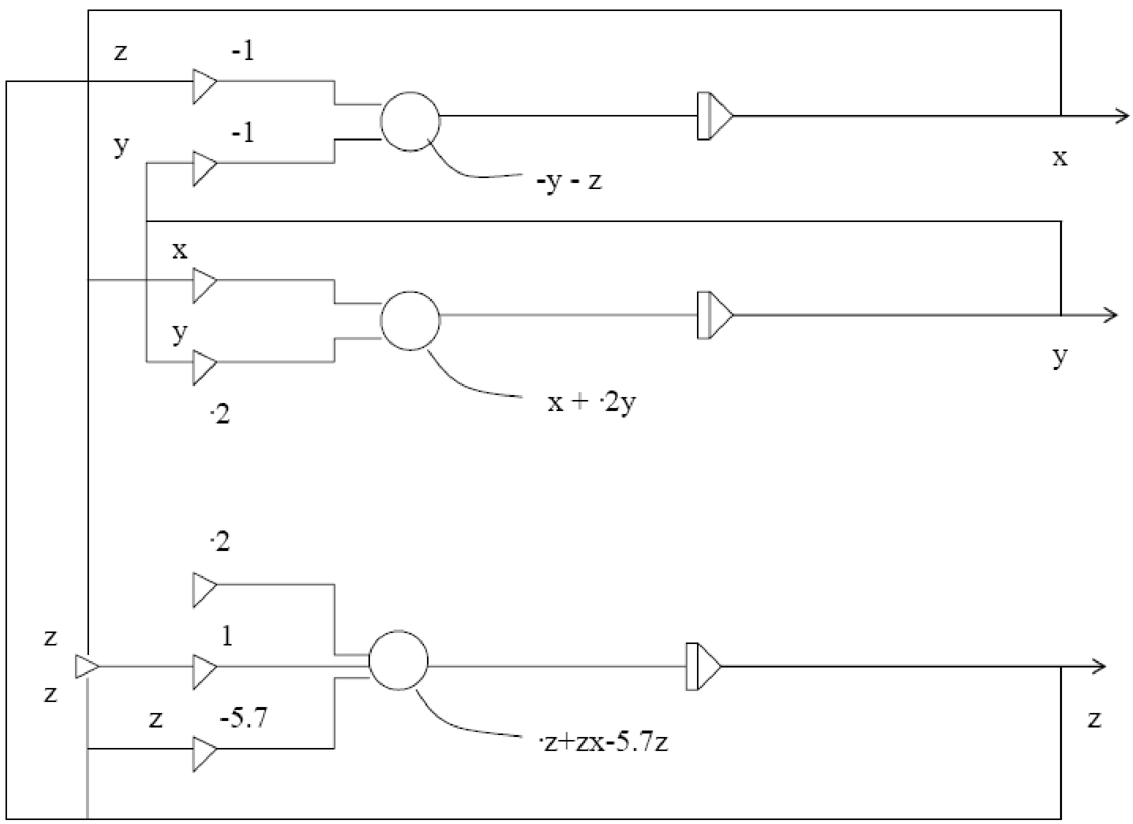

Figure 1: Rossler System

actual dynamics of the modelled economy? If this is so, the only problems an analog computing adherent needs to face are those that have been faced, and largely resolved, by the recursion theorist.

This is the reason for seeking the mathematical foundations of analog computation from the point of view of classical computability theory.

2 Theoretical Notes

> "Church's thesis, that all reasonable definitions of 'computability' are equivalent, is not usually thought of in terms of computability by a continuous computer, of which the general-purpose analog computer (GPAC) is a prototype."
>
> [13], p. 1011

Consider the linear, second order, differential equation that once formed the fountainhead of Keynesian, endogenous, macroeconomic theories of the cycle (indeed the kind of equation used in the Phillips differential equation model of the macroeconomy):

$$a\ddot{x} + b\dot{x} + kx = F \tag{1}$$

Solving, as in elementary textbook practice, for the second order term, $\ddot{x}$:

$$\ddot{x} = \frac{1}{a}F - \frac{k}{a}x - \frac{b}{a}\dot{x} \tag{2}$$

Integrating (2) gives the value for $\dot{x}$ to be replaced in the third term in the above equation.

Integrating $\dot{x}$, gives the value for x, and the system is 'solved'[2]. Thus, three

mechanical elements have to be put together in the form of a machine to implement a solution for (2):

- A machine element that would *add* terms, denoted by a circle;
- An element that could *multiply* constants or variables by constants, denoted by an equilateral triangle;

[2] This is the reason why analogue computing is sometimes referred to as resorting to 'bootstrap' methods. But it is more relevant, especially in computability contexts, to refer to this aspect as 'self-reference'. Recall Goodwin's perceptive observation, more than half a century ago:

> "A servomechanism *regulates its behaviour by its own behaviour* in the light of its stated object and therein lies the secret of its extraordinary finesse in performance. .... It is a matter of considerable interest that Walras' conception of and term for dynamical adjustment - tâtonner, to grope, to feel one's way - is literally the same as that of modern servo theory." (cf.[7] ; italics added)

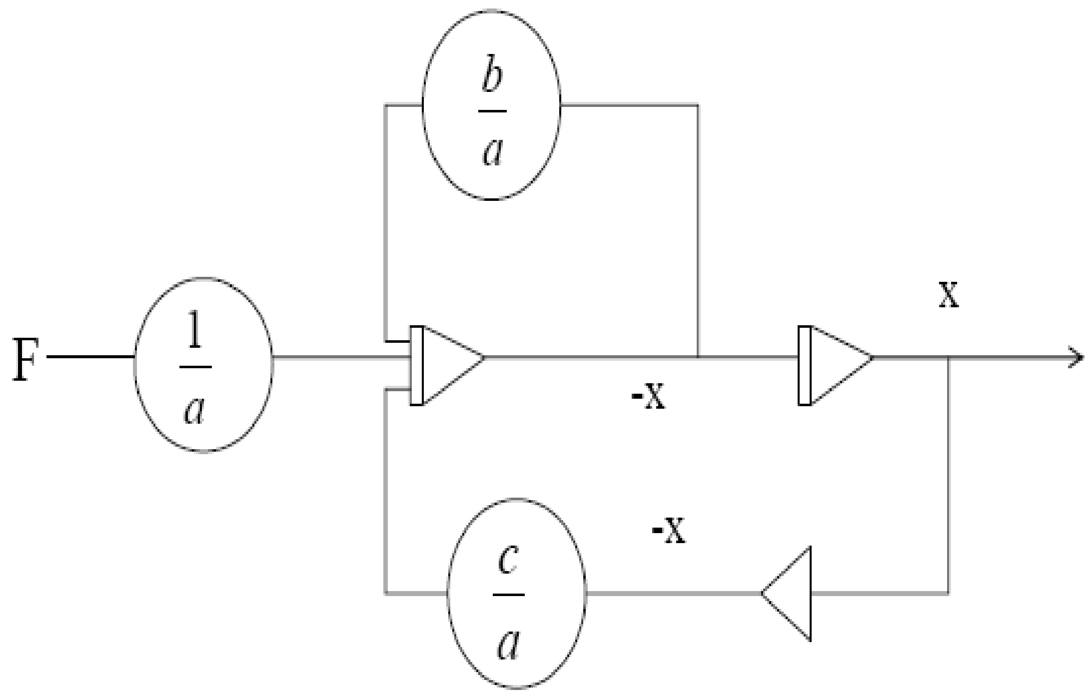

Figure 2: Linear Second Order Differential Equation

- An element that could '*integrate*', in the formal mathematical sense, without resorting to sums and limiting processes, denoted by a 'funnel-like' symbol;

One adder, three multipliers and two integrators, connected as in Figure 2, can solve the above equation[3]:

Note several distinguishing features of this analogue computing circuit diagram. First of all, there are no time-sequencing arrows, except as an indicator of the final output, the solution, because all the activity, the summing, multiplication and integration, goes on *simultaneously*. Secondly, no approximations, limit processes of summation, etc., are involved in the integrator; it is a natural physical operation, just like the operations and displays on the odometer

[3] One must add rules of interconnection such as each input is connected to at most one output, feasibility of feedback connections, and so on. But I shall leave this part to be understood intuitively and refer to some of the discussion in [11], pp. 9-11; observe, in particular, the important remark that (ibid, p.10, italics in the original):

> "*[F]eedback, which may be conceived of as a form of continuous recursion, is permitted.*"

in a motor car or the voltage meter reading in your home electricity supplier's measuring unit. Of course, there are the natural physical constraints imposed by the laws of physics and the limits of precision mechanics and engineering, something that is common to both digital and analogue computing devices, so long as *physical* realizations of mathematical formalisms are required.

In principle, any ODE can be solved using just these three kinds of machine elements linked appropriately because, using the formula for integrating by parts, a need for an element for differentiating products can be dispensed with. However, these machine elements must be supplemented by two other kinds of units to take into account the usual independent variable, time in most cases, and one more to keep track of the reals that are used in the adder and the multiplier. This is analogous to Turing's 'notes to assist the memory', but play a more indispensable role. Just as in Turing's case, one can, almost safely, conclude that 'these elements, appropriately connected, including "bootstrapping" - i.e., with feedbacks - exhaust the necessary units for the solving of an ODE'. Accepting this conjecture *pro tempore*, in the same spirit in which one works within the *Church-Turing Thesis* in *Classical Recursion Theory*, a first definition of an analogue computer could go as follows:

Definition 1 *A General Purpose Analogue Computer (GPAC) is machine made up of the elemental units: adders, multipliers and integrators, supplemented by auxiliary units to keep track of the independent variable and real numbers that are inputs to the machine process, and are interconnected, with necessary feedbacks between or within the elemental units to function simultaneously.*

Recalling the fertile and mutual interaction between partial recursive functions and Turing Machines, one would seek a definition, if possible by construction, of the class of functions that are analog computable by GPACs. These are precisely the *algebraic differential equations* ([11], p.7, [12], p.26, [15], pp.340-3).

Definition 2 *An algebraic differential polynomial is an expression of the form :*

$$\sum_{i=1}^{n} a_i x^{r_i} y^{q_{0i}} \left(y'\right)^{q_{1i}} .... \left(y^{(k_i)}\right)^{q_{k_i i}} \tag{3}$$

where a_i *is a* real *number*, $r_i, q_{0i}, ..., q_{k_i i}$ *are non−*negative integer *valued and* y *is a function of* x.

Definition 3 *Algebraic differential equations (ADEs) are ODEs of the form:*

$$P\left(x, y, y', y'', ..., y^{(n)}\right) = 0 \tag{4}$$

where P is an *algebraic differential polynomial* not identically equal to zero.

Definition 4 *Any solution $y(x)$ of an ADE is called differentially algebraic (DA); otherwise they are called transcendentally-transcendental ([12]) or hyper-transcendental ([15]).*

Clearly, the definition of ADEs includes all the usual sets of simultaneous systems of linear and nonlinear differential equations that economists routinely - and non-routinely - use. So, we are guaranteed that they are solvable by means of GPACs. Now one can pose some simple questions, partly motivated by the traditions of classical recursion theory:

- Are the solutions to ADEs, generated by GPACs, computable?
- Is there a corresponding concept to universal computation or a universal computer in the case of analogue computation by GPACs?
- Is there a fix point principle in analogue computing by GPACs that is equivalent or corresponds to the classic recursion theoretic fix point theorem?
- Is there a 'Church-Turing Thesis' for analogue computing by GPACs?

The reason I ask just these questions is that an economist who indiscriminately and arbitrarily formulates dynamical hypotheses in terms of ODEs and attempts to theorise, simulate and experiment with them must be disciplined in some way - in the same sense in which recursion theory and numerical analysis discipline a theorist with warnings on solvability, uncomputability, approximability, etc. It is all very well that the Bernoulli equation underpins the Solow growth model or the Riccati equation underpins the use of control theory modelling environments or the Rayleigh, van der Pol and Lotka-Volterra systems are widely invoked in endogenous business cycle theories. Their use for simulations calls forth the conundrums mentioned above for digital computers and they may require other kinds of constraints to be respected in the case of simulations by GPACs. There will, of course, be engineering constraints: precision engineering requirements on the constructions of the adders, multipliers and the integrators can only achieve a certain level of precision, exactly as the thermodynamic constraints of heat irreversibilities in the integrated circuits of the digital computer. I do not attempt to deal with these latter issues in this paper.

The answer, broadly speaking, to the first question is in the affirmative ([11], op.cit, §4, pp.23-27 and [13], Theorems 1 and 1', p.1012).

The answer to the second question is easier to attempt if the question is posed in a slightly different way, in tems of the relation between Turing Machines and Diophantine equations (cf. [9]).

Definition 5 *A relation of the form*

$$D(a_1, a_2, ..., a_n, x_1, x_2, ..., x_m) = 0 \tag{5}$$

where D is a polynomial with integer coefficients with respect to all the variables $a_1, a_2, ..., a_n, x_1, x_2, ..., x_m$ separated into *parameters* $a_1, a_2, ..., a_n$ and *unknowns* $x_1, x_2, ..., x_m$, is called a *parametric diophantine equation*.

A *parametric diophantine equation*, D, defines a set $\mathcal{F}$ of the parameters for which there are values of the unknowns such that:

$$\langle a_1, a_2, ..., a_n\rangle \in \mathcal{F} \iff \exists x_1, x_2, ..., x_m \left[D\left(a_1, a_2, ..., a_n, x_1, x_2, ..., x_m\right) = 0\right] \tag{6}$$

One of the celebrated mathematical results of the 20th century was the (negative) solution to *Hilbert's Tenth Problem [9].* In the eventual solution of that famous problem two crucial issues were: the characterisation of recursively enumerable sets in terms of parametric diophantine equations and the relation between Turing Machines and parametric Diophantine equations. The former is, for example, elegantly exemplified by the following result ([8], Lemma 2, p.407):

Lemma 6 *For every recursively enumerable set W, there is a polynomial with integer coefficients given by $Q\left(n, x_1, x_2, x_3, x_4\right)$, i.e., a parametric diophantine equation, such that, $\forall n \in \mathbb{N}$,*

$$n \in W \iff \exists x_1, \forall x_2, \exists x_3, \forall x_4 \left[Q\left(n, x_1, x_2, x_3, x_4\right) \neq 0\right] \tag{7}$$

The idea is to relate the determination of membership in a structured set with the (un)solvability of a particular kind of equation. If, next, the (un)solvability of this particular kind of equation can be related to the determined behaviour of a computing machine, then one obtains a connection between some kind of computability, i.e., decidability, and solvability and set membership. This is sealed by the following result:

Proposition 7 *Given any parametric Diophantine equation it is possible to construct a Turing Machine M, such that M will eventually halt, beginning with a representation of the parametric $n-tuple$, $\langle a_1, a_2, ..., a_n\rangle$ iff (16) is solvable for the unknowns $x_1, x_2, ..., x_m$.*

Suppose we think of ODEs as Parametric Diophantine Equations; recursively enumerable sets as the domain for continuous functions and GPACs as Turing Machines. Can we derive a connection between ODEs, continuous functions and GPACs in the same way as above? The affirmative answer is provided by the following proposition, which I shall call Rubel's Theorem:

Theorem 8 *(Rubel's Theorem): There exists a nontrivial fourth-order, universal, algebraic differential equation of the form:*

$$P(y', y'', y''', y'''') = 0 \tag{8}$$

where P is a homogeneous polynomial in four variables with integer coefficients.

The exact meaning of '*universal*' is the following:

Definition 9 *A universal algebraic differential equation* P *is such that any continuous function* $\varphi(x)$ *can be approximated to any degree of accuracy by a* C^∞ *solution,* $y(x)$*, of* P*. In other words:*

$$\forall \text{ positive continuous } \varepsilon(x),\ \exists y(x)\ s.t\ \mid y(x) - \varphi(x) \mid < \epsilon(x),\ \forall x \in (-\infty, \infty) \tag{9}$$

Recent developments (cf. [3],[2]) have led to concrete improvements in that it is now possible to show the existence of C^n, $\forall n, (3 < n < \infty)$; for example, the following is a specific Universal algebraic differential equation:

$$n^2 y'''' y'^2 - 3n^2 y''' y' + 2n(n-1) y''^2 = 0 \tag{10}$$

In this sense, then, there is a counterpart to the kind of universality propositions in classical recursion theory – computation universality, universal computer, etc., – also in the emerging theory for analogue computation, particularly, GPACs. Eventually, by directly linking such universal equations to Turing Machines via numerical analysis there may even be scope for a more unified and encompassing theory.

As for the third question, my answer goes as follows. GPACs can also be considered generalised fix-point machines! Every solution generated by a GPAC is a fixed-point of an ADE. This is a reflection of the historical fact and practice that the origins of fixed point theory lies in the search for solutions of differential equations, particularly ODEs[4].

Whether there is a Church-Turing Theses for analogue computation is difficult to answer. The reason is as follows. The concept of computability by finite means was made formally concrete after the notions of solvability and unsolvability or, rather, decidability and undecidability, were made precise in terms of recursion theory. These notions were made precise within the context of a particular debate on the foundations of mathematics - on the nature of the logic that underpinned formal reasoning. As Gödel famously observed:

[4] But also PDEs (partial differential equations), as George Temple pointed out ([17], p.119):

> "One of the most frutiful studies in topology has considered the mapping T of a set of points S into S, and the existence of fixed points x such that
>
> $$Tx = x$$
>
> The importance of these studies is largely due to their application to ordinary and partial differential equations which can often be transformed into a functional equation $Fx = 0$ with $F = T - I$ where $Ix = x$."

"It seems to me that [the great importance of general recursiveness (Turing's computability)] is largely due to the fact that with this concept one has for the first time succeeded in giving an absolute definition of an interesting epistemological notion, i.e., one not depending on the formalism chosen. In all other cases treated previously, such as demonstrability or definability, one has been able to define them only relative to a given language, and for each individual language it is clear that the one thus obtained is not the one looked for. For the concept of computability however, although it is merely a special kind of demonstrability or decidability[5] the situation is different. By a kind of miracle it is not necessary to distinguish orders, and the diagonal procedure does not lead outside the defined notion. This, I think, should encourage one to expect the same thing to be possible also in other cases (such as demonstrability or definability)."

[6], p.84.

So, to ask and answer an epistemological question such as whether there is a correspondence to the 'Church-Turing Thesis' in analogue computing by GPACs must mean that we must, first, characterise the formal structure and mathematical foundations of ODEs in a more precise way. I think this is an interesting methodological task, but cannot even be begun to be discussed within the confines of a simple expository paper such as this. I think, however, there will be an interplay between a logic on which continuous processes can be underpinned, say by Lukasiewicz's continuous logic, and the logic of ODEs[6]. My intuition is that there will be some kind of 'Church-Turing Thesis' in the case of analogue computing by GPACs and awareness of it will greatly discipline solution, simulation and experimental exercises by the use of GPACs (see also [13]).

3 Economic Modelling in the Analog Computing Mode

From the results tentatively outlined in section 2 it is clear – at least to me – that there is a duality between analog and digital computing. Since, in particular, there is a notion of 'universality' in analog and digital computing that are obviously 'dual' to each other, one can ask, more seriously than in the past, whether the 'economy can be simulated by a universal computer' and, if so, what kind of computer the Phillips Machine is. By this I mean to ask – and,

[5] I have always wondered whether this is not a misprint and the word that is meant to be here is not 'decidability' but 'definability'!

[6] I suspect that this will be fruitful link to pursue partly because Lukasiewicz, in the development of his continuous valued logic, abandons both the law of the excluded middle and proof by the method of *reductio ad absurdum* - both contentious issues in the debate between Hilbert and Brouwer that led to the foundational crisis in mathematics from which the work of Gödel and Turing emerged.

eventually, hopefully asnwer, the question whether the Phillips Machine is capable of universal computing or not. My immediate conjecture is that is it not capable of universal computing and is formally equivalent to a finite automatn, in the language of computability theory.

But the kind of questions, in greater detail, above and beyond the general question of the capability of universal computation, are similar to the ones posed by the redoubtable Richard Feynman, as a Physicist, to Physics:

First of all, Richard Feynman ([5], p.467) wondered:

> "Can physics be simulated by a universal computer?

Feynman, in his characteristically penetrating way, then asked three obviously pertinent questions to make the above query meaningful:

- What kind of physics are we going to *imitate*?
- What kind of *simulation* do we mean?
- Is there a way of *simulating* rather than *imitating* physics?

Before providing fundamental, but tentative, answers to the above queries, he adds a penetrating caveat (ibid, p.468; italics in original):

> "I want to talk about the possibility that there is to be an *exact* simulation, that the computer will do *exactly* the same as nature."

Feynman's answer to part of the first question was that the kind of physics we should simulate are 'quantum mechanical phenomena', because (ibid, p. 486):

> "...I'm not happy with all the analyses that go with just the classical theory, because nature isn't classical, dammit, and if you want to make a simulation of nature, you'd better make it quantum mechanical, and by golly it's a wonderful problem, because it doesn't look so easy."

But he was careful to point out, also, that there was a crucial mathematical difference between 'quantizing' and 'discretizing' (ibid, p. 488; italics added):

> "*Discretizing is the right word.* Quantizing is a different kind of mathematics. If we talk about *discretizing* ... of course I pointed out that we're going to *have to change the laws of physics.* Because the laws of physics as written now have, in the classical limit, a continuous variable everywhere ... ."

He was not the only giant in the natural sciences who wondered thus: Einstein, Schrödinger, Hamming, Toffoli, Fredkin and most recently, Penrose, too, have had speculative thoughts along similar lines. Einstein, in perhaps his last published work, seems to suggest that a future physics may well be in terms of the discrete:

> "One can give good reasons why reality cannot be represented as a continuous field. ...."
>
> [4], p.166

Roger Penrose, in his recently published, massive, vision of *The Road to Reality*, was even more explicit:

> [W]e may still ask *whether the real-number system is really 'correct' for the description of physical reality at its deepest level.* When quantum mechanical ideas were beginning to be introduced early in the 20th century, there was the feeling that perhaps we were now beginning to witness *a discrete or granular nature to the physical world* at its smallest scales.... Accordingly, various physicists attempted to build up an alternative picture of the world in which discrete processes governed all action at the tiniest levels. ....
>
> In the late 1950s, I myself tried this sort of thing, coming up with a scheme that I referred to as the theory of 'spin networks', in which the discrete nature of quantum-mechanical *spin* is taken as the fundamental building block for a *combinatorial* (i.e. a discrete rather than real-number-based) approach to physics."
>
> [10], pp.61-2; italics in the second paragraph as in original.

These speculations on the granular structure of 'reality' at some deep level arose out of purely theoretical developments in the subject, but in continuous interaction with the epistemology of measurement in well-designed and sound experimental environments. Where these reflections by Feynman, Einstein and Penrose leave the loose epistemology and wild methodological claims in [1], I am not at sure.

More importantly, the long-standing epistemological stance taken by those of us wedded to the philosophy of *Computable Economics*, that economic data is invariably and necessarily 'granular' and, therefore, computability theory or constructive mathematics is the correct approach for modelling computation and simulation of economic models seems to be validated by the results in the previous section. But the thorny problem of the intermediary role played by numerical methods and numerical algorithms in digital computing is, of course, circumvented by analogue computing. In this sense, there is a clear advantage, at least from an economic modelling point of view, in working with analog computers. But the cardinal message of Computable Economics is that the fundamental computing model for economics should be the Turing Machine. In that sense, the use of the Phillips Machine in economic modelling remains incomplete from an analytic point of view till it is shown, rigorously, that it is capable of universal computation.

References

[1] Blum, Lenore, Felipe Cucker, Michael Shub and Steve Smale (1998), **Complexity and Real Computation**, Springer Verlag, New York.

[2] Keith M. Briggs (2000), *Another Universal Differential Equation*, **tantalum.bt-sys.bt.co.uk:tex/universal-ode.tex**, April 13, 2000.

[3] R.J.Duffin (1981), *Rubel's Universal Differential Equation,* **Proceedings of the National Academy of Sciences USA**, Vol. 78, No.8, August, pp. 4661-2.

[4] Einstein, Albert, (1955), '*Relativistic Theory of the Non-Symmetric Field*', in: **The Meaning of Relativity** (5th Edition), Appendix II, pp. 133-166, Princeton University Press, Princeton, New Jersey.

[5] Feynman, Richard P, (1982), '*Simulating Physics with Computers*', **International Journal of Theoretical Physics**, Vol. 21, Nos. 6/7, pp. 467-488.

[6] Kurt Gödel [1946, (1965)], *Remarks Before the Princeton Bicentennial Conference on Problems in Mathematics*, in **The Undecidable: Basic Papers on Undecidable propositions, Unsolvable Problems and Computable Functions** edited by Martin Davis, Raven Press, New York, pp. 84-88.

[7] Richard M. Goodwin (1951), *Iteration, Automatic Computers, and Economic Dynamics*, **Metroeconomica**, Vol. III, Fasc. 1, April, pp. 1-7.

[8] James Jones (1981), *Classification of Quantifier Prefixes over Diophantine Equations*, **Zeiechrift für Mathematische Logik und Grundlagen der Mathematik**, Vol. 27, pp. 403-10.

[9] Yuri V. Matiyasevich (1993), **Hilbert's Tenth Problem**, The MIT Press, Cambridge, Massachusetts.

[10] Penrose, Roger, (2004), **The Road to Reality: A Comprehensive Guide to the Laws of the Universe**, Jonathan Cape, London.

[11] Marian Boykan Pour-El (1974), *Abstract Computability and Its Relation to the General Purpose Analog Computer (Some Connections Between Logic, Differential Equations and Analog Computers)*, **Transactions of the American Mathematical Society**, Volume, 199, November, PP. 1-28.

[12] Lee A. Rubel (1988), *Some Mathematical Limitations of the General-Purpose Analog Computer*, **Advances in Applied Mathematics**, Vol. 9, pp.22-34.

[13] Lee A. Rubel (1989), *Digital Simulation of Analog Computation and Church's Thesis*, **Journal of Symbolic Logic**, Vol. 54, No.3, September, pp. 1011-17.

[14] Tim Sauer and James A. Yorke (1991), *Rigorous Verification of Trajectories for the Computer Simulation of Dynamical Systems*, **Nonlinearity**, Vol. 4, pp. 961-79.

[15] Claude E. Shannon (1941), *Mathematical Theory of the Differential Analyzer,* **Journal of Mathematics and Physics**, Vol. XX, No. 4, December, pp. 337-354.

[16] Smale, Stephen, (1991), *Theory of Computation*, pp. 60-9, in: **Mathematical Research Today and Tomorrow: Viewpoints of Seven Fields Medalists!**, edited by C.Casacuberta and M. Castellet, Springer-Verlag, Berlin and Heidelberg.

[17] George Temple (1981), **100 Years of Mathematics**, Gerald Duckworth & Co. Ltd., London.

[18] Weihrauch, Klaus (2000), **Computable Analysis: An Introduction**, Springer-Verlag,Berlin & Heidelberg.