

Unity of science as seen through the universal computer

JOSÉ FÉLIX COSTA*

*Department of Mathematics, Instituto Superior Técnico, Universidade de Lisboa
Centro de Filosofia das Ciências da Universidade de Lisboa, Portugal*

Key words: Empirical laws; Induction; Learning theory; Philosophy of science; Scientific theory; Theory unification

Unification is a common word in Physics as well as in other sciences such as Biology (with the case of natural selection). In this paper, working with basic concepts such as models of empirical scientist that after collecting finitely many experimental observations, comes with a law, and theoretical scientist that after collecting finitely many scientific laws, comes up with a theory, we apply concepts of algorithmic learning theory to shed some light on the concepts of unity and unification common in the philosophy of science, namely that theories (as collections of theorems) established by inductive methods may not be unifiable by inductive methods.

1 INTRODUCTION

Algorithmic learning theory is in general introduced by induction quizzes such as sequences of number associations: $\langle 9, 72 \rangle$, $\langle 5, 20 \rangle$, $\langle 7, 42 \rangle$, $\langle 8, 56 \rangle$, $\langle 6, 30 \rangle$, $\langle 3, ? \rangle$. To complete this sequence, we attempt to find an algorithm or postulate a “mathematical law” from a set of admissible “mathematical laws” such as those described by computable functions. A similar activity was done by scientists like Robert Boyle while measuring the pressure of a gas against the volume it occupies, in a sequence of lab experiments on gases kept at constant temperature. A possible answer to the numerical quiz is the function $n \mapsto n \times (n - 1)$ (giving $\langle 3, 6 \rangle$); similarly, Robert Boyle succeeded in finding

* email: fgc@math.tecnico.ulisboa.pt

the physical law “pressure \times volume = constant” (of the specific gas). Data for the numerical quiz is a sample of points possibly from the graph of the function $n \mapsto n \times (n - 1)$. Data for Robert Boyle’s quiz is a sample of points possibly from the graph of the function volume \mapsto constant/pressure, for some value of the constant.

Conventional scientists write mathematical formulas using some grammar of symbolic expressions. In this text, the scientist expresses the law by means of a computer program. That this idea has some support is witnessed by the fact that all the empirical relations and most of the solutions to differential equations so far handled by physicists are representable by the concept of a computable function.

In this paper, we apply mathematical tools to prove that inductive methods operating on different set or function domains cannot, in general, be unified. In particular, scientific theories established by inductive methods on collections of empirical laws may not be unifiable in a sense that will be made precise.

The organization of this paper is the following:

Concepts from computability are provided in Section 2. There we discuss theoretical possible realities that are either certain laws of Nature, modeled by recursive functions, or theories, modeled by recursive enumerable sets. We really doubt that any law so far established by science escapes computability, but no one really knows the Physics of the future.*

Scientists or scientific methods produce intelligible hypotheses. As we will discuss in Section 3, our hypotheses are either empirical laws or theories and are conjectured either by humans or by computer programs.

The data available to scientists or scientific methods are either finite subsets of the graphs of empirical laws, i.e. finite input/output pairs, or finite lists of empirical laws considered as theorems of some theory to be. Sections 4, 5, and 6 introduce the basic building blocks of empirical law or theory identification. Precision of scientific data will also be considered in Section 4.

Finally, in Sections 7 and 8, we discuss the limitations of the scientific method and a putative disunity of science by means of mathematical theorems.

* However, the computable character of the physical law does not imply that all physical systems are simulable. We can compute Newton’s law of universal gravitation as relation between measurable concepts, but a composite system obeying Newton’s law is not necessarily simulable. In a sequence of papers, Warren Smith provided some amount of evidence that Physical theories such as Quantum Mechanics and General Relativity are simulable (see [25, 26]).

This paper is conceptually self-contained.

Important remark: The use of Physics as example in this text could be replaced by any other empirical science that gives predictions.

2 COMPUTABILITY

Consider any common programming language \mathbb{X} (such as the well known language C). Consider all programs P that we can write having as input a number $^\dagger n$ and as an output another number $\psi(n)$. In theoretical computer science, with generality, abstract objects are encoded as numbers given as inputs to programs P : a set, or a list of (natural, integer, or rational) numbers can be encoded into a single number; the same applies to finite graphs and other structures of visual mathematics; computer programs written in \mathbb{X} can also be encoded as numbers. With the encoding procedures, we can input a number n into a program P that first decodes n , let us say, into the code m of another program P' and an input k to P' . To sum up: everything that can be considered for programming can be encoded into the natural numbers.

Programs in \mathbb{X} (\mathbb{X} -programs for short) with input n eventually halt or run forever. As a crucial statement in computability we have the *undecidability of the halting problem*: there is no program H written in \mathbb{X} such that, given a number, decoded into a \mathbb{X} -program P of code m and an input number k to P , H outputs 1 if P halts on k and 0 if P does not halt on k .

We define a *partial recursive function* ψ as the full collection of pairs $\langle n, \psi(n) \rangle$, i.e. input/output, of some program P , such that P halts on each input n of the collection providing $\psi(n)$ as output. For the remaining values of n the function is said to be undefined. A partial recursive function is also called a computable function. If the collection contains a pair for each value of $n \in \mathbb{N}$, then we say that the function ψ is *recursive*. The set of all recursive functions is denoted by \mathcal{R} . If there exists a program P for a recursive function ψ containing only FOR loops (of the kind FOR $k := a$ TO b DO), then the function is said to be *primitive recursive*. If the FOR loops are nested up to a priori fixed number, then the function is said to be *elementary*. Computing recursive functions that are not primitive recursive requires WHILE loops implementing unbounded search procedures.

Consider now programs P with output 1 or 0. We define a *recursively enumerable* set S (r.e. S for short) as the collection of inputs for which a

[†] By *number* we refer to a natural number (denoted by the symbol \mathbb{N}), unless otherwise specified.

program P generates output 1. If, besides this property, either P or another 1-equivalent \mathbb{X} -program P' [‡] answers 0 to all the inputs not in S (meaning that P' halts for all inputs), then we say that S is *recursive*. In this case P' is said to be a *decision procedure* for S . If ψ is a partial (or total) function computable by the program P , then by $dom(\psi)$, the *domain* of ψ , we denote the set of input values for which P halts. This set is recursively enumerable. It can also be proved that each recursively enumerable set coincides with the domain of some computable function.

A set (either recursive or just recursively enumerable) defined by an \mathbb{X} -program P of code e is denoted by W_e . A function defined through an \mathbb{X} -program P of code e is denoted by ϕ_e . Note that W and ϕ are conventionally fixed symbols and only the code e changes.

Sometimes it is useful to consider also programs for functions ψ in \mathbb{X} with two input numbers m, n and output $\psi(m, n)$, instead of encoding m and n into a single number. There are two fundamental results to recall (see Rogers [24]).

First, the so-called s–m–n theorem states that, if the value of m is fixed yet arbitrary, then we can find a computable function g such that $g(m)$ provides a number code of a program for $\psi(m, n)$ now as a function of n with m fixed. Since m is arbitrary, we get program codes $g(0), g(1), g(2), \dots$ for $\psi(0, n), \psi(1, n), \psi(2, n), \dots$, respectively. Using the symbol ϕ , we may write $\psi(m, n) = \phi_{g(m)}(n)$.

Second, there is an important theorem due to Stephen Kleene that states that there exists a number e coding for an \mathbb{X} -program P , such that P on input n outputs the value of $\psi(e, n)$. This is a subtle statement, because e appears both as datum and program code. As an example, Kleene's theorem can be applied to prove that the set of recursive functions ψ such that $\psi(0)$ is the code of an \mathbb{X} -program for ψ itself is not empty. This collection of functions constitutes the set SD of *self-describing* functions that will be considered in Section 7.2.

Let us see how to do it.

From any recursive function f (for which we know that an \mathbb{X} -program P exists) we will build a new function $g \in SD$. Consider the auxiliary function h of two inputs such that, for every numbers m, n ,

$$h(m, n) = \begin{cases} m & \text{if } n = 0 \\ f(n - 1) & \text{if } n > 0 \end{cases} .$$

[‡] I.e., P and P' output 1 exactly with the same inputs.

That that function h is computable is witnessed by the following informal \mathbb{X} -program P' : on input the natural numbers m and n , P' tests n for 0 first; if $n = 0$, then P' outputs the number m that it received as input together with n ; else, if $n > 0$, then P' runs the existing subroutine P , testifying that ψ is computable, on input $n - 1$. Now, we see that Kleene's theorem applied to h states that there exists a number e such that, on input e and n , P' behaves as the \mathbb{X} -program of code e on input n . Consequently, we have that $\phi_e \in SD$, where $\phi_e(n) = h(e, n)$.

Finally, a few words about the complexity of programs. By counting assignments or comparisons, or both, during an \mathbb{X} -program execution, we can describe the evolution of the computation. Each assignment or comparison is called a *step*. Let P be an \mathbb{X} -program either for a set or for a function, n an input, and i, j numbers. We write $P(n) \downarrow_j$ to say that P with input n halts in less than $j + 1$ steps and $P(n) \uparrow_j i$ to say that P with input n outputs i in less than $j + 1$ steps.

It is difficult to suggest an empirical law in Physics that is not primitive recursive. However, we will be working with the set of recursive functions that strictly includes the primitive recursive functions. Thinking on laws that are expressed by means of differential equations, a wide spectrum of numerical techniques over the real numbers are analyzed from a complexity point of view in [2].

3 THE SCIENTIST CONCEPT

Learning an empirical law in the sciences (as Physics), is comparable to solving a quiz: e.g., given instances of a functional association as $\langle 9, 72 \rangle$, $\langle 5, 20 \rangle$, $\langle 7, 42 \rangle$, $\langle 8, 56 \rangle$, $\langle 6, 30 \rangle$, $\langle 3, ? \rangle$, we question whether we can extend this sequence in a consistent way. The same question arises with the sequence $\langle 1 + 4, 5 \rangle$, $\langle 2 + 5, 12 \rangle$, $\langle 3 + 6, 21 \rangle$, $\langle 8 + 11, ? \rangle$.

As a working example, we suggest Robert Boyle who solved a similar quiz in 1662 by performing a sequence of lab experiments of measuring the volume of some confined amount of air against the pressure exerted on it.[¶] When Boyle was confronted with his own data, reproduced in Table 1, he came about with the law

$$pressure \times volume = constant ,$$

where the value of the constant depends on the specific ideal gas undergoing an isothermal process.

[¶] Edme Mariotte discovered the same law in 1679.

An account of learning laws or grammar rules either from observations or by listening to native language speakers was formalised by Gold in 1967 in [12]. More abstract developments of Gold's idea were done in the late seventies by Blum and Blum in [1] and Case and Smith in [6] (as an extension of [5]). From that time on, recursion-theoretical learning theory was established as a specialization in algorithms.

Volume (V)	Pressure (P)	PV	Volume (V)	Pressure (P)	PV
1.0	29.750	29.750	10.0	3.000	30.000
1.5	19.125	28.688	12.0	2.625	31.500
2.0	14.375	28.750	14.0	2.250	31.500
3.0	9.500	28.500	16.0	2.000	32.000
4.0	7.125	28.500	18.0	1.875	33.750
5.0	5.625	28.125	20.0	1.750	35.000
6.0	4.875	29.125	24.0	1.500	36.000
7.0	4.250	29.750	28.0	1.375	38.500
8.0	3.750	30.000	32.0	1.250	40.000
9.0	3.375	30.375			

Table 1
Boyle's original data (from Magie 1935). This table provides an idea of what experimental data is for functions in the work of an empiricist).

There is not much ontological differences between solving the quiz and solving the observations in a lab, up to some expected observation errors that we will ignore. The quiz $\langle 9, 72 \rangle, \langle 5, 20 \rangle, \langle 7, 42 \rangle, \langle 8, 56 \rangle, \langle 6, 30 \rangle, \langle 3, ? \rangle$ can be imagined as a sample of points from the graph of the function $\langle n, n(n-1) \rangle$. (The reader conjectures that the answer to the quiz is $\langle 3, 6 \rangle$.) Boyle's quiz is given as a sample of the graph of the function $\langle volume, pressure \rangle$. An empiricist would be able, in principle, to identify a set of such laws from an infinite number of possible mathematical relations. Moreover, an empiricist identifies the law after having observed a *finite sample/sequence* of the infinite graph of the relation, no matter the order of observations. More observations can either corroborate or refute the law (according with Popper methodology), and in the latter case produce a theory change, that is a reformulation of the previous law. Persistently, if the law is not complex enough, the empiricist will end up learning the law in the limit, after sufficient enough number of experiments and observations.

There is much in common between small children learning grammar and scientists learning the laws of Nature. Namely, both children and the scientists are exposed only to positive instances. Parents, in general, do not provide negative instances such as “this sentence is incorrect”; negative instances such as manifestations of “levitation is forbidden” is not provided by Nature.

Conventional scientists write mathematical formulas on paper using some conventional universal grammar. Alternatively, the scientist may write an equivalent computer program, provided either as common text or as a number code. We will consider encodings of programs written in some programming language \mathbb{X} into the natural numbers (\mathbb{N}). The encoding techniques are quite common in computer science (see [10] for the recursive functions). Specific encoding techniques for scientific laws and Physics are discussed in [28]. In this later paper, the author provides several encoding functions for the rationals and the laws of Physics conceived in a discrete world of some precision.

We assume that empirical laws are either recursively enumerable sets or recursive functions and that scientific methods are partial recursive functions. This is the *computationalist hypothesis* (see the book of Kevin Kelly [16] and also the introductory article of John Case [4]). E.g., method \mathcal{M} , on inputting the observations from an experiment (such like the looking-up table of $\langle \text{volume, pressure} \rangle$ of a contained gas), outputs a computer program P that simulates the experiment, informally described as follows: on input of the values assigned to some variables (as the *volume*), P outputs the predicted value of other variables (as the *pressure*). Assuming that a law by its own nature refers to a recursive relation, we accept the following methodology: *a law written in standard fashion and a computer program are interchangeable*. We refer to the computer program Boyle as a *scientist* or as a *scientific method*.[§]

4 EMPIRICIST AND THEORIST

Information for empiricists is provided in experimental notes, scientific articles, etc. E.g., information to the scientist Boyle is provided as a table for a function:

0	1	2	3	4	...
$\langle 29.750, 1.0 \rangle$	$\langle 7.125, 4.0 \rangle$	$\langle 3.750, 8.0 \rangle$	$\langle 2.625, 12.0 \rangle$	$\langle 2.000, 16.0 \rangle$...

where the numbers in the top line provide some (arbitrary) order to the experimental data registered in the bottom line, as seen in Table 1.

[§] A good introduction to automated scientific discovery is the book by Pat Langley et al. [18].

A text T for a function is a map from numbers to pairs of numbers (n , function value at n). By $T[t]$ we denote the sequence of the first t pairs of the text T , from the 0th to the $(t - 1)$ th pair, i.e. $T[t] = T(0)T(1) \cdots T(t - 1)$, where $T(i)$ is the pair of numbers at position i in the sequence. Let

$$SEG = \{T[n] : T \text{ is a text for a function and } n \text{ is a number}\},$$

be the set of prefixes for recursive functions and $INIT \subset SEG$ be the subset of prefixes of texts for functions $\psi \in SEG$ in increasing order of the independent variable, as $\#\psi(0)\#\psi(1)\#\cdots\#\psi(t-1)\#$, where $\#$ is a separation symbol between numbers.^{||} For each $\sigma \in SEG$, $content(\sigma)$ provides the corresponding set of pairs in σ . Since σ is a prefix for a function, no such pairs $\langle m, n_1 \rangle$ and $\langle m, n_2 \rangle$, with $n_1 \neq n_2$, may belong to $content(\sigma)$. The sequence σ can also be seen as a partial function from numbers to numbers, denoted by $\hat{\sigma}$, defined as

$$\hat{\sigma}(m) = \begin{cases} n & \text{if } \langle m, n \rangle \in content(\sigma) \\ \text{undefined} & \text{otherwise} \end{cases}.$$

The information for theorists is provided by scientific articles, books, etc. If we think in a theory of Physics such as Maxwell's Electromagnetic Theory, we realize that the theoretical physicist, instead of looking at a quiz of numbers (such as Robert Boyle did), he looks at samples of empirical laws such as Coulomb's law of electrostatics, Ampère's law for electric circuits, Faraday's law of induction, etc., and, possibly by adding some new hypothesis (such as the new concept of displacement current), he formulates a theory, herein seen as a recursively enumerable collection of theorems. The scientist instead of a quiz of numbers or relations is confronted with a quiz of "theorems", or rather "theorems to be", and conjectures the code of recursively enumerable set, e.g. the set of theorems induced by some axiomatization.

A text T for a set is a map from numbers (ordering) to numbers (encodings of empirical laws). Again, $T[t] = \#T(0)\#T(1)\#\cdots\#T(t-1)\#$. We will be using the set SEQ of prefixes of texts for sets. If T is a text for a set, then by $content(T)$ we denote the set of numbers in T .

Definition 1 (Scientific method, Gold [12]) *A scientist or scientific method is a computable function either of type $SEQ \rightarrow \mathbb{N}$ (for sets) or of type $SEG \rightarrow \mathbb{N}$ (for functions).*

^{||} Note that, in this case, the ordering number of the pairs is implicit:

$$\begin{array}{c|c|c|c|c|c} 0 & 1 & 2 & 3 & 4 & \dots \\ \langle 0, \psi(0) \rangle & \langle 1, \psi(1) \rangle & \langle 2, \psi(2) \rangle & \langle 3, \psi(3) \rangle & \langle 4, \psi(4) \rangle & \dots \end{array}$$

We give now an idea how an empirical law could be algorithmically established. Assume that the scientist believes that the empirical relations are primitive recursive.

Let us assume for simplicity that we are learning the constant π given by its decimal expansion obtained from successive measurements by means of successively more sophisticated instruments. After a sufficiently long but finite sequence 3, 3.1, 3.14, 3.141, 3.1415, ... of approximations of π , the scientist knows that such a number is the theoretical π , a number that has a primitive recursive n -digit.[#]

Everytime the suitable scientific method receives a new approximation, it outputs the same or a new conjecture, that is (the code of) a computer program that, for each number n , computes π_n , the n th-digit of π . Suppose that the method \mathcal{M} behaves in the following way: for each finite sequence of approximations of π , \mathcal{M} successively decodes the sequence of natural numbers 0, 1, 2, 3, ..., into a sequence of \mathbb{X} -primitive recursive programs, until it finds the first conjecture consistent with the input data seen thus far (see Algorithm 1). Soon or later, it will happen that for some number p , the \mathbb{X} -program of code p reproduces the digits of the unknown π so far read. This p will be the conjecture of the scientist at that point. Further input approximations, will probably make \mathcal{M} change conjecture, one, twice, three times... until ...

Definition 2 (Convergence of a scientist, Gold [12]) *A scientist or scientific method \mathcal{M} converges to a \mathbb{X} -program code p on text T for a set or for a function if, for all but finitely many numbers k , $\mathcal{M}(T[k]) = p$. A scientific method \mathcal{M} identifies a text T if there is an \mathbb{X} -program code p such that \mathcal{M} converges to p on T and p generates content(T).*

Definition 3 (Identification of sets, Gold in [12]) *A scientist \mathcal{M} $TxtEx$ -identifies a recursively enumerable set L if \mathcal{M} identifies every text for L . A scientist \mathcal{M} $TxtEx$ -identifies a class of sets \mathcal{L} if \mathcal{M} $TxtEx$ -identifies every set L from \mathcal{L} .*

Definition 4 (Identification of functions, Gold in [12]) *A scientist \mathcal{M} Ex -identifies a (total) computable function ψ (= recursive function ψ) if \mathcal{M} identifies every text for ψ . A scientist \mathcal{M} Ex -identifies a set of functions Ψ if \mathcal{M} identifies every function ψ from Ψ .*

[#] We can say that a number ν , or a constant of Physics, is a recursive number if the n th digit of its decimal expansion is given by some recursive function. The digits of π are given by a primitive recursive function.

Algorithm 1: Scientist \mathcal{M} identifies the class of primitive recursive functions

```

Scientist  $\mathcal{M}(\#\psi(0)\#\psi(1)\#\psi(2)\#\dots\#\psi(n-1)\# : SEG) : \mathbb{N}$ 
  Variable  $i, k \in \mathbb{N}$ ;
   $i \leftarrow 0$ ;
   $k \leftarrow 0$ ;
  While  $k < n$ 
    IF  $\phi_i(k) = \psi(k)$ 
      Then
         $k \leftarrow k + 1$ ;
      Else
         $k \leftarrow 0$ ;
         $i \leftarrow i + 1$ 
      End
    Conjecture  $i$ ;
  End

```

Identification of sets by computable scientists is called *TextEx*-identification and identification of functions is called *Ex*-identification. No one knows, at a particular time, if convergence is already achieved. *Ex* comes from **Ex**plain and *Text* from **Text**. Note that both concepts are limit concepts. Although this search for a code serves the purpose of learning all primitive recursive functions,** that in our view includes the majority of known empirical laws of all sciences, the convergence process is not good enough and the conjectures produced are not satisfactory, for long prefixes of text are needed until the convergence to a final hypothesis is met. However, it works.

In what follows, *TextEx* is the collection of all *TextEx*-identifiable classes of sets and *Ex* is the class of all *Ex*-identifiable sets of functions.

There is a remarkable (mathematical) difference between identification of an empirical law and the identification of a theory. As discussed in Section 7.1, if Nature is not sufficiently complex, then all laws can be learned in the limit. However, for theories, a guarantee of success in the limit is provided only for very elementary sets of theorems. ††

By rescaling and encoding the values of physical magnitudes, we can define physical laws as relations between natural numbers. In a world of experimental error, convergence can be addressed with a different definition:

** Or even an enlargement of the set of primitive recursive functions, yet strictly contained in the set of recursive functions.

†† Set, language, or theory identification are formal discussed in philosophical grounds in the paper on Gold's theorem by Kent Johnson [15].

Definition 5 (*Scientific success on a single function*) We say that the scientist \mathcal{M} identifies $\psi \in \mathcal{R}$ if there exists an $e \in \mathbb{N}$ and numbers $p, \ell \in \mathbb{N}$ such that, for $t \geq p$, $\mathcal{M}(\psi[t]) = e$ and, for all $t \in \mathbb{N}$, $|\overline{\phi_e(t)} - \overline{\psi(t)}| \leq 2^{-\ell}$, where the line over the functions means the decoding of natural numbers into rational numbers.^{‡‡}

5 TOTAL METHODS

We now discuss whether scientists or scientific methods can be considered total maps.

Definition 6 We say that a scientific method \mathcal{M} is total on a recursive function ψ if the method \mathcal{M} provides a conjecture for all time t , where time is the number of pairs (with possible repetitions) in the input prefix σ of the graph of ψ . We say that a scientific method \mathcal{M} is total on a set S of recursive functions if \mathcal{M} is total on each function of S . Finally, \mathcal{M} is total if \mathcal{M} is total on the whole set of recursive functions \mathcal{R} .

We provide a proof of two very basic facts about scientific methods as in Definition 2.

Proposition 1 For each scientific method \mathcal{M} for functions, there exists another scientific method \mathcal{N} for functions, algorithmically obtainable from \mathcal{M} , such that: (a) \mathcal{N} is total and (b) if \mathcal{M} identifies any recursive function ψ , then \mathcal{N} also identifies ψ .

For the proof, we consider any method \mathcal{M} , assuming that it identifies any given recursive function $\psi \in S$, and we algorithmically specify a total method \mathcal{N} that does the same.

Let T be any text for ψ . On input observations $T[t]$, the method \mathcal{N} calls the old method \mathcal{M} to run for t steps, successively and orderly, on all the prefixes of $T[t]$, e.g. $T[0] = \epsilon$, $T[1] = T(0)$, $T[2] = T(0)T(1)$, $T[3] = T(0)T(1)T(2)$, ..., $T[t] = T(0)T(1)T(2) \cdots T(t-1)$. Then the method \mathcal{N} takes the longest prefix $\tilde{\sigma}$ of $T[t]$ (the latest one) for which \mathcal{M} outputs a conjecture in t steps (see end of Section 2). In case that no conjecture were produced in t steps for any prefix, the new method \mathcal{M} outputs 0, otherwise the method outputs $\mathcal{M}(\tilde{\sigma})$. Method \mathcal{N} is total. The algorithm is synthesized in Figure 1.

^{‡‡} In the standard context of learning theory, we take $\ell = +\infty$, and we have \mathcal{M} converging to e on $\psi[t]$ and, for all $t \in \mathbb{N}$, $\phi_e = \psi$.

```

 $\mathcal{N}$  :
Function  $\mathcal{N}(\sigma : SEQ) : \mathbb{N}$ ;
Begin
   $t := |\sigma|$ ; % Size of data
  Simulate  $t$  steps of method  $\mathcal{M}$  on successively data  $\sigma[0], \sigma[1], \sigma[2], \dots, \sigma[t]$ ;
  If any of the runs  $\mathcal{M}(\sigma[0]), \mathcal{M}(\sigma[1]), \mathcal{M}(\sigma[2]), \dots, \mathcal{M}(\sigma[t])$  halt in  $t$  steps
    Then Return the latest conjecture
  Else Return 0
End

```

Figure 1
Method \mathcal{N} constructed from the possibly partial method \mathcal{M} .

Let us now suppose that T is a text for a recursive function ψ . Then, there exists an order p such that method \mathcal{M} , on input $T[p]$, converges to the final conjecture and there exists a number $q \geq p$ such that, for $j \geq q$, j steps are enough for \mathcal{M} to converge on $T[p]$. Thus, analysing data of size greater or equal to q , \mathcal{N} outputs the final conjecture on all prefixes of text T of size greater or equal to q .

In general, a text for a function is not given in increasing order of the independent variable, i.e. text in the canonical form $\#\psi(0)\#\psi(1)\#\psi(2)\#\dots$ (standing for $\langle 0, \psi(0) \rangle \langle 1, \psi(1) \rangle \langle 2, \psi(2) \rangle \dots$), although any pair $\langle i, \psi(i) \rangle$ for a small i occurs in any text for ψ , even though at a large distance from the first pair in the text. As the next theorem states, any scientific method for canonical text can be generalized to a scientific method for arbitrary texts.

Proposition 2 *Let \mathcal{M} be a method that Ex -identify the recursive function ψ . If \mathcal{M} converges to the conjecture e on the canonical text for ψ , then there exists a method $\tilde{\mathcal{M}}$ that converges to e on all texts for ψ .*

Let T be a canonical text for ψ and $\tilde{\mathcal{M}}$ be the method that, on input prefix σ of text T , computes first the longest prefix $\tilde{\sigma} \in INIT$ such that all pairs in $\tilde{\sigma}$ are also in σ and then calls \mathcal{M} on $\tilde{\sigma}$. Note that such prefixes can always be formed even if they require very long input sequences σ . Note also that, as the size of σ grows towards infinity, both σ and $\tilde{\sigma}$ become texts for ψ , being $\tilde{\sigma}$ canonical. Then, since \mathcal{M} Ex -identify ψ , we conclude that, if \mathcal{M} converges to e on the canonical text for ψ , then the scientist $\tilde{\mathcal{M}}$ converges to e on all texts for ψ .

The former reasoning that allows the scientist to provide a method for data in increasing order of the independent variable cannot be done for sets. Text for sets are just list of numbers separated by the symbol #. Recursive enumerable sets cannot in general be enumerated in ascending order — only recursive sets can. However, Fulk in [11] proved that if a class of r.e. sets is *TextEx*-identifiable, then a scientist can be specified that produces the same final conjecture, no matter the text provided for each such set.

From this point on, *we will be consider only scientific methods that are total maps. Moreover, scientific methods for functions will be operating only on canonical text.*

6 RATIONALITY IN QUESTION

Algorithmic learning opens the door to the foundations of formal scientific inquiry in what concerns the limits of algorithmic cognition and many aspects of rationality of the scientific method. Several cognition strategies have been discussed and cataloged in groups since the sixties. As an example, we discuss consistency, a problem that was addressed by the Blums in [1].

The Blums required that at any time, conjectures should explain data so far obtained: if \mathcal{M} is a consistent method for a function ψ , then given text T for ψ , the successive conjectures $\mathcal{M}(T[0])$, $\mathcal{M}(T[1])$, $\mathcal{M}(T[2])$, ..., $\mathcal{M}(T[n])$ should be able to reproduce the information $T[0] = \#$, $T[1] = \#\psi(0)\#$, $T[2] = \#\psi(0)\#\psi(1)\#$, ..., $T[n] = \#\psi(0)\#\psi(1)\#\dots\#\psi(n-1)\#$, respectively, given as input. Moreover, we say that \mathcal{M} is consistent in a set of laws S if \mathcal{M} is consistent with each law ψ in S . Then, a statement apparently against rationality can be proved: that consistent methods have strictly less inductive power than those that may not be consistent along the process of inquiry (until identification is achieved). In other words, consistent algorithmic methods identify strictly less functions than general, possibly non-consistent methods.^{¶¶}

7 UNITY

We now question whether there is unity among scientific methods as introduced in Definition 1. The answer is straightforward: it depends on how

^{¶¶} A more provocative way of stating this result is the following: if there are natural laws populating the whole Turing universe, then there are laws that cannot be discovered by consistent inductive methods.

computationally complex the empirical laws are. In Sections 7 and 8, we will analyze what algorithmic learning theory can say about unity of inductive method. Two main results can be called for such a discussion. The first is the negative statement on the unity of method for functions and the second is the non-existence of universal method that covers the universe of recursively enumerable theories. Although these statements are known in specialized literature in learning theory, we provide detailed soft proofs. We first examine how elementary a common empirical law can be.

7.1 How elementary is a law of Nature?

If we consider the class of all empirical relations between measurable concepts that can be derived from current scientific theories, we may wonder whether a scientific method exists, according with Definitions 1, 2, and 4, that is capable of identifying them all, given sufficient long histories of observations.

The answer to this question relies on the complexity of relations that can be found in the natural sciences. Some empirical relations are direct algebraic relations between quantities, such as Galileu's, Kepler's, or Ohm's laws, etc., whereas some others come from the solution of differential equations, such as Newton's second law.

Since, in our case, relations between measurable concepts are to be provided by suitable \mathbb{X} -programs, we discuss first how expressive \mathbb{X} is as a language. The maximum computational power of an imperative \mathbb{X} -routine is achieved only when we allow arbitrary nested WHILE loops (with syntax "WHILE halting condition DO"), i.e. loops such that the number of their steps is eventually unpredictable and are interrupted only when some condition becomes false (see Section 2). Commonly, differential equations from Physics can be solved by iterative methods that have the number of their steps bounded by exponentials on the number of digits of precision.

As we saw in Section 2, primitive recursive functions can always be implemented by means of FOR loops. If we allow an arbitrary number of compositions of nested and sequential FOR loops, we get a collection of \mathbb{X} -programs (and therefore of primitive recursive functions) that halt for every input. The exact correspondence of primitive recursive functions and this class of \mathbb{X} -programs was given for the first time by Meyer and Ritchie in [19]. As a consequence, the primitive recursive functions can be enumerated and encoded into the natural numbers.

All primitive recursive functions are *Ex*-identifiable in the sense of Definitions 1, 2, and 4 by a scientific method that consists of a methodical and

orderly examination of all possible \mathbb{X} -programs that are made of arbitrary nested FOR loops, by successive decodings of the natural numbers $0, 1, 2, \dots$ and providing as output the first conjecture consistent with the input data/measurements done thus far (see Section 4). If the relation is primitive recursive, sooner or later a code number e will match the data. This method just shows that a “black box” exists that on imputing arbitrary long yet finite sequence of data of each empirical relation of some family, it outputs, after an arbitrary long but finite number of steps, the code of a scientific law compatible with it.

It is difficult to say whether any empirical relation can be described by primitive recursive functions over the rational numbers. If true, then the empirical laws are identifiable in the limit and, since a single known method suffices, that there is unity in empirical science.

Larger classes of functions are *Ex*-identifiable by a single scientist, but the set of all recursive functions (which is larger than any *Ex*-identifiable extension of the primitive recursive functions) is not, due to the nonunion theorem of Section 7.2. In general, subclasses of the class \mathcal{R} of recursive functions cannot be identified in the limit by single general routines.

7.2 Nonunion theorem

The nonunion theorem is the first important limiting result of algorithmic learning theory after Gold, appearing in the first paper by the Blums (see [1]). Essentially, it states that arbitrary scientific methods, according to Definitions 1, 2, and 4, may not be unifiable in a single method. Scientific method \mathcal{E} aims at explaining observations/measurements relative to domain of phenomena E and scientific method \mathcal{F} aims at explaining observations/measurements relative to domain of phenomena F . In the end of the day, a scientist attempts to explain the observations/measurements of both worlds E and F by means of scientific method \mathcal{A} . Scientific method \mathcal{A} can be viewed as method unification.

However, for a sufficiently complex reality there is no unity in method. It would seem that one could just say “if the observed phenomena is in domain E , then use \mathcal{E} , else, if it is in domain F , then use \mathcal{F} ”. This seems to be an unified method since it applies to both $\mathcal{E} \cup \mathcal{F}$. Blums’s theorem states that such a putative method does not always exist since it involves non-decidable predicates.

Theorem 1 (Nonunion theorem; Gold [12]; Blum and Blum [1]) *Classes $TextEx$ and Ex are not closed under union.*

Recalling Definitions 1, 2 and 4 this theorem can be rephrased as

Theorem 2 (Nonunion theorem) *If a method \mathcal{E} either Ex or $TextEx$ -identifies a domain E and another method \mathcal{F} either Ex or $TextEx$ -identifies a domain F , does not necessarily exist a method that either Ex or $TextEx$ -identifies the domain $E \cup F$.*

We prove this theorem for the case of Ex -identification. The easier case of $TextEx$ -identification is left to the reader's curiosity. We first provide two sets of functions that are Ex -identifiable by easy methods:

1. *Subject of inquiry:* Identification of functions in the set AEZ , i.e. recursive functions ψ that are **A**lmost **E**verywhere **Z**ero. *Scientific method:* On input the prefix $\#\psi(0)\#\psi(1)\#\dots\#\psi(t-1)\#$, the scientist builds the ordered list μ of non-zero values $\psi(i)$ and outputs the law (its code) ^{§§}

If $x \in \text{dom}(\hat{\mu})$ **Then** $\hat{\mu}(x)$ **Else** 0 ,

i.e.: if the input value of x occurs in first place of a pair $\langle x, y \rangle$ in μ , then the output is y , otherwise the output is 0. After a sufficiently long input prefix $\#\psi(0)\#\psi(1)\#\dots\#\psi(t-1)\#$ of a text for a function in AEZ all values of the function are zero, the remaining elements of the list μ become constant, and the scientist converges to the same program code of the function ψ .

2. *Subject of inquiry:* Identification of functions in the set SD of **S**elf-**D**escribing functions, i.e. functions ψ such that the first value of ψ , $\psi(0)$, is a programming code for ψ (as seen in Section 2). *Scientific method:* On input $\#\psi(0)\#\psi(1)\#\dots\#\psi(t-1)\#$ the scientist outputs $\psi(0)$. If the text for ψ is not canonical, then the scientist waits to read $\psi(0)$, that will appear soon or later in the list of observations, and outputs the same value.

In terms of scientific theories, nonunion means that although two different scientific methods can account for all observable laws in the two different domains, no scientific method can generalize both. In fact, no recursive scientific method can be found that Ex -identifies all the functions in $AEZ \cup SD$.

Notice that, although the property that defines SD is not expected in the nature of the laws of Physics, no one knows exactly how complex the laws of Physics are.

^{§§} See Section 4 for the meaning of $\hat{\mu}$.

Herein, the proof method consists in finding a single counterexample of union of scientific methods. Most generally, with this single counterexample comes a countable infinite family of counterexamples just by changing the subjects of inquiry. Expectedly, more reasonable instances of nonunion of method can be found, surely a bit harder to introduce and discuss.

<pre> FUNCTION ψ : Function $\psi(e, x : \mathbb{N}) : \mathbb{N}$; Var $y : \mathbb{N}$; $\sigma : \text{list of } \mathbb{N}$; % σ is the prefix of ψ Begin $\sigma := \langle 0, e \rangle$; % Value of $\psi(e, 0)$ Loop Find the lexicographically least continuation τ of σ, such that $\mathcal{M}(\tau) \neq \mathcal{M}(\sigma)$; σ becomes τ; If the pair (x, y) occurs in σ for some y Then Return y End Loop End </pre>

Figure 2

If \mathcal{M} is a method for AEZ , then, no matter the size of prefix σ , there exists a lexicographically minimal string $\tau \supset \sigma$ that makes \mathcal{M} change his mind. In such a way, a total function can be constructed by means of Kleene's recursion theorem.

The proof of the nonunion theorem runs as follows.

To prove that there is no method for $AEZ \cup SD$, we prove that no scientific method \mathcal{M} capable of Ex -identifying AEZ can Ex -identify SD . For the purpose, we look for a suitable function ψ in SD , such that no scientific method for AEZ can distinguish the prefixes of ψ from prefixes of AEZ functions.

Let us look closer into the argument. We choose a function $\psi \in SD$ such that $\psi(0)$ is some value to be fixed and $\psi(n)$, for $n > 0$, is either 0 or 1. Given a sequence of (canonical) observations σ of ψ , $\sigma = \langle 0, e \rangle \langle 1, \psi_1 \rangle \cdots \langle n-1, \psi_{n-1} \rangle$, where e is the code of ψ and each value $\psi_1, \psi_2, \dots, \psi_{n-1}$ is 0 or 1, there is always a lexicographically strictly longer prefix of observations of ψ , $\tau \supset \sigma$, such that \mathcal{M} conjectures differently on τ and σ ; otherwise scientist \mathcal{M} would not be distinguishing streams with common prefix σ , such like $\sigma \langle n, 0 \rangle$ and $\sigma \langle n, 1 \rangle$, although both are prefixes of texts for different functions in AEZ .

The *informal* routine in Figure 2 describes a procedure of making the intended SD function out of AEZ . Now, the parameter e , that is also the value of function $\psi(e, x)$ at 0, can be wisely chosen for the purpose: by Kleene's theorem, just fix e as the code of $\psi(e, x)$ itself! Function $\psi(e, x)$, having e fixed, belongs to SD and it is not Ex -identifiable by \mathcal{M} since, according to its own programming code in Figure 2, it would have to instantiate an infinite number of theory changes as seen in the clause $\mathcal{M}(\tau) \neq \mathcal{M}(\sigma)$. In this way, SD (containing ϕ_e) is not Ex -identifiable by \mathcal{M} . Since \mathcal{M} is arbitrary within the methods for AEZ , we conclude that $AEZ \cup SD$ is not Ex -identifiable.

8 UNIVERSAL METHOD

In this section we discuss theory achievement, assuming that the scientist applies some inductive method to sequences of expected theorems of the theory to be, i.e. from a number of given empirical laws (theorems to be), an axiomatization is attempted, from which a recursively enumerable set of theorems follows. In the case of Maxwell's electromagnetic theory, from Maxwell's axioms it follows the empirical integral laws of Coulomb, Ampère and Faraday, the existence of electromagnetic waves (Maxwell), the laws of reflection and refraction of light (Fresnel), etc.^{||||}

We concentrate now on sets (recursively enumerable sets of encodings of theorems) instead of functions and we redefine the concept of scientific method in order to incorporate different compartments of expertise. Let us admit that, when we shift attention from a domain to another, we apply different cognitive procedures.^{##} We introduce the parameterized method that allows a change of "compartment" of knowledge: from subject i to subject j , method changes from $\Gamma[i]$ to $\Gamma[j]$ (note the square brackets). Abstractions $\Gamma[i]$ and $\Gamma[j]$ are scientific methods according with Definition 1.

Definition 7 *A parameterized scientific method Γ is any recursive function from numbers to scientific methods (or scientists).*

The number i is a parameter such that $\Gamma[i]$ is candidate scientific method for the class of recursively enumerable sets W_i , i.e. the index i encodes a

^{||||} Currently some attempts in axiomatizing some fragments of Physics have been made such as [20, 22, 27]. However, the idea of axiomatic format as in Bunge [3] will do in what follows.

^{##} E.g., with respect to visual procedures, we know that that identification of a face and identification of a color are done by different modules of the visual cortex ($V4$ and $V4\alpha$, respectively); the same with the identification of form and identification of motion ($V1$, $V2$, $V3$ and $V5$, respectively), etc. Semir Zeki calls this feature "disunity of consciousness" (see [29]). This disunity exists for other brain processes, as discussed in [30].

procedure to enumerate the set W_i , such that each number $j \in W_i$ should be considered an index of a recursively enumerable set, for a theory. In decoding the number i into an \mathbb{X} -program, we are able to enumerate the codes of \mathbb{X} -programs for possibly different recursively enumerable sets. In this sense, i provides the code of a class of sets. (A more general framework on parameterized methods can be found in [13, 14].) Thus, a parameterized scientific method Γ is a hypermethod that generates methods $\Gamma[0]$, $\Gamma[1]$, $\Gamma[2]$, etc. E.g., we could say that $\Gamma[i]$ identifies theories consistent with electromagnetic phenomena, $\Gamma[j]$ identifies theories that are consistent with gravitational phenomena, etc. Learning theorists see this parameter as communication: we moved to subject i , so that we now think in i mode”.

We prove that, in the general case, no such universal method can exist unless the compartments of knowledge are simple enough.

Definition 8 *We say that a parameterized scientific method Γ performs a set (a family of subjects of inquiry) X if, for all $i \in X$, $\Gamma[i]$ *TextEx-identifies* W_i . We say that X is learnable if some parameterized scientist performs X .*

If we interpret a recursively enumerable set as being a collection of theorems, a class of sets as being a collection of theories, then a single number just provides a class of theories. A parameterized scientific method is some function of identifiers and finite sequences of theorems such that, once receiving a communication i , provides a method for the class of sets indexed by i . In a sense, it works like a student who applies inductive methods for first understanding electromagnetic theory, and thereafter applies different inductive methods to understand classical or quantum mechanics. Then, no “computationally universal student” can exist that is able by communication of context to identify everything that is computationally identifiable. The proof of non-existence of a universal learner is based in a proof for a particular case that implies the general case (see [13]).

Proposition 3 *The class of numbers i such that W_i contains exactly two numbers (two indexes of r.e. sets) is not learnable.*

At a first look this putative parameterized method Γ would seem simple enough: on receiving index i , the method $\Gamma[i]$ should be able just to distinguish between two sets!

Proposition 4 *The class \mathcal{W} of numbers i such that W_i is *TextEx-identifiable* is not learnable.*

Let us suppose that \mathcal{W} is learnable. It follows that all its subsets are learnable, in particular the class of numbers i such that W_i contains exactly two numbers is learnable. But such a fact is in contradiction with Proposition 3.

We now provide our proof of Proposition 3 recalling that each finite class of recursively enumerable sets is *TextEx*-identifiable.

Let p be a fixed code number for the set \mathbb{N} ($W_p = \mathbb{N}$) and e a code number for some subset of \mathbb{N} . We consider any total parameterized method Γ and prove that $\Gamma[i]$ cannot *TextEx*-identify $W_i = \{e, p\}$, for some number i that is a computable function g of e , i.e. $i = g(e)$. The existence of function g is justified as follows: we start from the computable function

$$f(e, n) = \begin{cases} 1 & \text{if } n = e \text{ or } n = p \\ \text{undefined} & \text{otherwise} \end{cases}$$

and we apply the s-m-n theorem to guaranty the existence of a function g (see Section 2) such that $g(e)$ provides the code of an \mathbb{X} -program for the function f when given parameter e and input n . The domain of f is exactly the set $\{p, e\}$ that we denote $W_{g(e)}$.

Now we construct a suitable number e .

The recursive function ψ in Figure 3 calls for the scientific method $\Gamma[g(e)]$. We choose e (different from p) such that $\phi_e(x) = \psi(e, x)$, with domain W_e , according with Section 2. Examining closely the function $\psi(e, x)$, we conclude that either (a) $\psi(e, x) = 1$ for all numbers $x = 0, 1, 2, 3, \dots$, or (b) $\psi(e, x) = 1$ for $x = 0, 1, \dots, m$, for some m , and it is undefined for all the other numbers x greater than m ; case (a) occurs when both internal loops halt for all inputs x and case (b) occurs when one of the two internal loops does not halt for some input x . The limit sequence of σ is in any case a text for W_e . We show that the scientific method $\Gamma[g(e)]$ fails to identify $\{W_e, W_p\}$ proving the statement of the theorem.

In the case (a) above, for the suitable e satisfying Kleene's theorem, the set of values x such that $\psi(e, x) = 1$, i.e. W_e , is the set of natural numbers (and, therefore, e is another index for the natural numbers); it is easy to see that, in the limit, σ is a text T for the natural numbers (indeed a fat text, since each number occurs infinitely many times). However, method $\Gamma[g(e)]$ fails to identify this set given this particular text T , since it is forced to change conjecture infinitely often as seen in clause **Until**. Notice that appending n , in the definition of σ after the second loop, guarantees that, in the limit, the text will cover the natural numbers even in the case that the dovetail procedure for i and j keeps repeating the same numbers and do not append all the natural numbers to σ .

```

FUNCTION  $\psi$  :
Function  $\psi(e, x : \mathbb{N}) : \mathbb{N}$ ;
Var  $i, j, n : \mathbb{N}; \sigma_0, \sigma : \text{list of } \mathbb{N}$ ;

Begin
   $n := 0$ ;
   $\sigma := \#$ ;
  Loop forever
    Begin
      Search for  $i$  and  $j$  such that  $\Gamma[g(e)]$  on input  $\sigma\#0\#1\#\dots\#i\#$  conjectures
        a program code that converges to ‘yes’ in  $j$  steps, for a set of inputs
        that strictly contains  $\text{content}(\sigma) \cup \{0, 1, \dots, i\}$ ;
       $\sigma := \sigma\#1\#\dots\#i\#$ ; % Text being extended
      If  $x \in \text{content}(\sigma)$  Then Return 1; % At this point  $W_e$  is a finite set
       $\sigma_0 := \sigma$ ;
      Repeat  $\sigma := \sigma\#$  Until  $\Gamma[g(e)](\sigma_0) \neq \Gamma[g(e)](\sigma)$ ;
        % Repeat adding ‘nothing’ until the condition is met
       $\sigma := \sigma\#n$ ;
       $n := n + 1$ 
    End
  End

```

Figure 3
Construction of a doubleton of sets that scientist $\Gamma[g(e)]$ is not able to identify.

Undefinedness of $\psi(e, x)$ is due to an infinite internal loop. Either the dovetail procedure for i and j fails, meaning that the parameterized method with parameter $g(e)$ cannot identify $W_p = \mathbb{N}$, or the **Repeat** loop does not terminate and the method is unable to identify the finite set W_e since, given successive prefixes of a text for W_e , $\sigma, \sigma\#, \sigma\#\#, \sigma\#\#\#, \sigma\#\#\#\# \dots$ either it did not converge properly or it did converge to a code of a *superset* of $W_e = \text{content}(\sigma_0)$ (forced by the condition “a set strictly containing $\text{content}(\sigma) \cup \{0, 1, \dots, i\}$ ” in the search loop, meaning that the loop only halts when a further number not in $\text{content}(\sigma) \cup \{0, 1, \dots, i\}$ is given a ‘yes’ by program code $\Gamma[g(e)](\sigma\#0\#1\#\dots\#i\#)$ after j steps, for some i and j).

We conclude that the arbitrary parameterized method Γ , given parameter $g(e)$, does not identify the doubleton $W_{g(e)} = \{W_e, W_p\}$. Thus, no universal method exists for sets X that contain the doubletons of theories.

Note again that Proposition 3 implies Proposition 4. Proposition 4 applies to domains with an arbitrary number of theories. We see that, even with two theories in the same compartment, the scientist either might not be able to identify one of them or to distinguish between the two, changing his mind infinitely often. The same result applies to any number of theories in each compartment.

9 CONCLUSIONS

The expressive power of recursive functions is apparent in modern science, when a computer program, expressing the evolution of a physical system, takes the place of differential equations. Physical systems tend to be described by computer programs characterized by emergent properties that are not derived from the properties of their components, but rather from their complex interference. Instead of a differential equations, systems are explained and forecast by a computer programs implementing explanatory top-down rules. Such systems can exhibit complex behavior that cannot be described by elementary or even primitive recursive functions. Computer programs performing such simulations replace the standard physical law.

We focused on Physics as an activity both experimental and theoretical and we considered that inductive methods are also recursive, avoiding the conjecture of Roger Penrose (see [23]) and the strong AI community (see [17], namely papers 6 and 7). Thus, we assumed that inductive methods in science are computable and can generate recursive empirical laws from experimental data and recursively enumerable theories from empirical laws. We showed that a large collection of potential empirical laws are learnable by a single inductive method and we conjectured that no empirical predictive law has ever escaped to be primitive recursive. In fact, we do not know any genuine candidate for a scientific law at escapes computability, although there are some attempts to create some (see [7, 8, 9]). Not accepting that reality is computable places the world beyond the Turing limit were we have “computations” that cannot be specified by finite means: computation without a program and law without an algorithm.

Thinking further in the development of a theory of Physics as Maxwell’s Electromagnetic Theory (see [21]), we advanced the idea that the theoretical physicist, instead of looking at a quiz of numbers (as the empirical scientist seems to do), looks at a quiz of theorems to be, i.e. of empirical facts, and establishes a theory. We then discussed the disunity of science in terms of compartments of scientific knowledge, suggesting that, if Nature is suffi-

ciently complex, no universal learner, can inductively establish a general theory from finite sequences of theorems to be (taking altogether the empirical evidences from the different domains).

In a future paper we will address the question of inductive inference of a scientific theory based on some common logic background.

Acknowledgements. The research of José Félix Costa is supported by Fundação para a Ciência e Tecnologia, projeto FCT I.P.:UID/FIL/00678/2013.

REFERENCES

- [1] Lenore Blum and Manuel Blum. Toward a mathematical theory of inductive inference. *Information and Control*, 28:125–155, 1975.
- [2] Lenore Blum, Felipe Cucker, Michael Shub, and Steve Smale. *Complexity and Real Computation*. Springer, 1998.
- [3] Mario Bunge. *Foundations of Physics*, volume 10 of *Springer Tracts in Natural Philosophy*. Springer-Verlag, 1967.
- [4] John Case. Algorithmic Scientific Inference: Within Our Computable Expected Reality. *International Journal of Unconventional Computing*, 8(3):192–206, 2012.
- [5] John Case and Carl Smith. Anomaly hierarchies of mechanized inductive inference. In *Proceedings of the tenth annual ACM symposium on Theory of computing*, pages 314–319. ACM, 1978.
- [6] John Case and Carl Smith. Comparison of identification criteria for machine inductive inference. *Theoretical Computer Science*, 25(2):193–220, 1983.
- [7] Barry Cooper and Piergiorgio Odifreddi. Incomputability in Nature. In Barry Cooper and Sergei Goncharov, editors, *Computability and Models, Perspectives East and West*, University series in mathematics, pages 137–160. Springer, 2003.
- [8] Barry S. Cooper. Mathematics, metaphysics and the multiverse. In M. J. Dinneen, B. Khoussainov, and A. Nies, editors, *Computation, Physics and Beyond 2012*, Lecture Notes in Computer Science, pages 252–267. Springer, 2012. International Workshop on Theoretical Computer Science, WTCS 2012, Dedicated to Cristian S. Calude on the Occasion of His 60th Birthday, Auckland, New Zealand, February 21–24, 2012.
- [9] Barry S. Cooper. Turing’s titanic machine? *Communications of the ACM*, 55(3):74–83, 2012.
- [10] Nigel Cutland. *Computability*. Cambridge University Press, 1980.
- [11] M. Fulk. Prudence and other conditions on formal language learning. *Information and Computation*, 85:1–11, 1990.
- [12] E. M. Gold. Language identification in the limit. *Information and Control*, 10:447–474, 1967.
- [13] Sanjay Jain, Daniel N. Osherson, James S. Royer, and Arun Sharma. *Systems That Learn. An Introduction to Learning Theory*. The MIT Press, second edition, 1999.
- [14] K. Jantke. Natural properties of strategies identifying recursive functions. *Elektronische Informationverarbeitung und Kybernetik*, 15:487–496, 1979.

- [15] Kent Johnson. Gold's theorem and cognitive science. *Philosophy of Science*, 71:571–592, 2004.
- [16] Kevin T. Kelly. *The Logic of Reliable Inquiry*. Oxford University Press, 1996.
- [17] L. J. Landay and J. G. Taylor (editors). *Concepts for Neural Networks, A Survey*. Springer, 1998.
- [18] Pat Langley, Herbert A. Simon, Gary L. Bradshaw, and Jan M. Zytkow. *Scientific Discovery, Computational Explorations of the Creative Processes*. The MIT Press, first edition, 1987.
- [19] Albert Meyer and Dennis Ritchie. The complexity of loop programs. In *Proceedings of the 22nd National Conference*, pages 465–470. Thompson Book Company, 1967.
- [20] Attila Molnár and Gergely Székely. Axiomatizing relativistic dynamics using formal thought experiments. *Synthese*, 192(7):2183–2222, 2015.
- [21] Margaret Morrison. *Unifying Scientific Theories: Physical Concepts and Mathematical Structures*. Cambridge University Press, 2007.
- [22] Bruno Woltzenlogel Paleo. Physics and proof theory. *Applied Mathematics and Computation*, 219(1):45 – 53, 2012. Towards a Computational Interpretation of Physical Theories Towards a Computational Interpretation of Physical Theories.
- [23] Roger Penrose. *Shadows of the Mind*. Oxford University Press, 1994.
- [24] Hartley Rogers, Jr. *Theory of Recursive Functions and Effective Computability*. MIT Press, third edition, 1987. First edition published in 1967 by McGraw Hill.
- [25] Warren Smith. Church's thesis meets quantum mechanics, 1999. Published online at <http://www.math.temple.edu/~wds/homepage/churchq.ps>
- [26] Warren Smith. Church's thesis meets the N-body problem. *Applied Mathematics and Computation*, 178(1):154–183, 2006.
- [27] Gergely Székely. Logic and relativity theory. *Synthese*, 192(7):1937–1938, 2015.
- [28] Matthew P. Szudzik. The computable universe hypothesis. In Hector Zenil, editor, *A Computable Universe Understanding and Exploring Nature as Computation*, pages 479–523. World Scientific, 2012.
- [29] Semir Zeki. *Inner Vision, An Exploration of Aet and the Brain*. Oxford University Press, 1999.
- [30] Semir Zeki. The disunity of consciousness. *Trends in Cognitive Science*, 7:214–218, 2003.