ON DISCOVERING SCIENTIFIC LAWS

José Félix Costa
Department of Mathematics
Instituto Superior Técnico, University of Lisbon, Portugal
Centro de Filosofia das Ciências da Universidade de Lisboa, Portugal
e-mail: fgc@math.tecnico.ulisboa.pt

Abstract

We review how inductive methods operate on recursive data to uncover empirical laws as those of Physics, as well as the research programme of weakening learning criteria aiming at identifying the class of recursive functions. We think that this work should be better known, namely as of applications to understand large scale limitations of scientific discovery. Methods of recursion theoretic learning theory are also revised to cope with difficulties in the cases of infinite self-reference of identification theory.

There are two main concerns in this paper. The first is to convince the reader that the set of recursive relations is a good model of the universe of potential empiric laws. The second is to make the point that any such law can potentially be discovered by an unconventional scientist that accepts to weaken the criteria of what is "to know a law".

This paper can be read by any patient reader with some experience in programming.

1 Introduction and motivation

An empirical law is quite generally an algebraic relation of some simplicity. In fact, as far as we know, no one has found in the entire history of science an empirical relation that, abstracting from the type of the constants and variables, cannot be expressed by an elementary relation. Thus, saying that the universe of recursive relations — those relations that can be expressed by computers with the elementary relations occupying the lowest level of complexity — is too restrictive to express Nature (up to fixed precision concrete measurements) sounds ridicule. In this paper we will make the point that no other relations can be learned from Nature.

In some sciences, like Physics, laws get two different formulations, either they have a microscopic or local formulation, generally introduced by a differential equation, or they have a macroscopic description. Electromagnetic Theory of Maxwell follows both descriptions in a dialog between microscopic and macroscopic presentations. Both microscopic and microscopic laws have syntaxes obeying to simple rules. From one we can get the other, e.g., from a microscopic law we can get the macroscopic counterpart by numerical integration (or just integration). In this case, the computer programs might look heavier than those representing elementary relations and we can discuss if they

are still placed in lower levels of the recursive relations. In [6, 4, 5] we prove that a large family of differential equations can be numerically be solved in the lowest levels of the recursive relations. We can even discuss if there is a law to be that escapes to the recursive formulation. Of course, such laws cannot be used to predict, a fundamental aspect of a model. In fact, as said above, they cannot even be formulated.

Scientific theories have been build, but not always, by abduction from a set of empirical facts. These laws are relations between measurements and have been expressed either by algebraic relations or by differential equations. The previous paragraph suggests that all these laws can be inferred in the limit from observations. This statement may appear unconvincingly since, although the relation / law is known in finitely many points, there are still infinitely many other points where the relation can possibly be arbitrarily defined. We will see that cannot be the case for all the recursive functions. If the natural laws are sufficiently simple, then, in a sense that we will make precise bellow, all the laws can be learned after finitely many observations and expressed in a conventional way such like Kepler's third law of planetary motion, Boyle's law of ideal gases, Galileo's law of free fall, Ohm's law, Coulomb's law of electrical forces, etc. As the natural laws become complex, we may need unconventional ways to express them, in a way that only rarely occurred in the Sciences.

Conventional scientists write mathematical formulas on paper using some conventional universal grammar. Alternatively, the scientist can 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 into the natural numbers ($\mathbb{N}$). The encoding techniques are quite common in computer science. They can be easily understood. Suppose that we want to encode all the possible written English texts (such like computer programs) into the natural numbers. First we order the texts by size, i.e. the number of symbols in each text, then, for each size, we order the texts in the alphabetic order of character occurrence in the text. Then the first text receives number 0, the second number 1 and so forth. [1]

Learning an empirical law in the sciences (as Physics), is comparable to solving a quiz: e.g., given instances of a relation as $(1 \oplus 4, 5)$, $(2 \oplus 5, 12)$, $(3 \oplus 6, 21)$, $(8 \oplus 11, ?)$, we question whether we can extend this sequence in a consistent way. As a working example, we suggest Galileo's similar quiz in his book of 1638 (*Discourses on Two New Sciences*), that by performing a sequence of lab experiments in inclined planes using rolling balls, established the laws governing velocity and distance as functions of time. When Galileo was confronted with his own data, reproduced in Table 1, she came about with the law

$$distance = \alpha \times time^2 \ .$$

An account of learning free fall law from a table, or the grammar rules of our native language, either from observations, or by listening to native language

[1] Specific encoding techniques for scientific laws and Physics are discussed in [21]. In the former paper, the author provides several encoding functions for the rationals and the laws of Physics conceived in a discrete world of some precision.

speakers, was formalised by Daniel Gold in 1967 in [12]. More abstract developments of Gold's idea where done in the late seventies by Lenore Blum and Manuel Blum in [2] and John Case and Karl Smith in [10] (as an extension of [9]) inter alia. From those times on, recursion-theoretical learning theory was established as a specialization in algorithms.

Time (T)	Distance (D)	D/T	D/T^2
0.1	0.098	0.98	9.8
0.2	0.392	1.96	9.8
0.3	0.882	2.94	9.8
0.4	1.568	3.92	9.8
0.5	2.450	4.90	9.8
0.6	3.528	5.88	9.8

Table 1: Idealized data obeying the law of uniform acceleration (see Magie 1935, pp. 456 – 472, or Pat Langley, Herbert A. Simon, Gary L. Bradshaw and Jan M. Zytkow 1987, pp 11 in [17]).

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 $(1 \oplus 4, 5)$, $(2 \oplus 5, 12)$, $(3 \oplus 6, 21)$, $(8 \oplus 11, ?)$ can be imagined as a sample of points from the graph of the function $(m + n, m(n - 1))$. (The reader conjectures that the answer to the quiz is $(8 \oplus 11, 80)$.) Galileo's quiz is given as a sample of the graph of the function $(distance, time)$. 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 Karl Popper's 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 but finite number of experiments and observations.

There is much in common between 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; negative instances such as manifestations of "levitation is forbbiden" is not provided by Nature.

We assume that empirical laws are recursive relations and that scientific methods are recursive relations as well. This is the *computationalist hypothesis* (see the book of Kevin Kelly [16] and also the introductory article of John Case [8]). E.g., method $\mathcal{M}$, on input the observations from an experiment (such like the looking-up table of (time, distance) in a free fall experiment), outputs a computer program "Galileo" that establishes the law: on input of the values assigned to some variables (as the *time*), Galileo outputs the predicted value of other variables (as the *distance*). Assuming that a law by its own nature refers

to a recursive relation, we accept that *a law written in standard fashion and a computer program are interchangeable*. We refer to the computer program "Galileo" as a *scientist* or as a *scientific method*. [2]

The point of Philosophy of Science that we will be discussing in this paper too is that inductive methods have identified very elementary laws, let us say elementary relations.

2 Computability

Consider any common programming language $\mathbb{X}$ (such as the well known language JAVA). Consider all programs P that we can write having as input a number [3] n and as 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 digital pictures such has 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'. We write $n = \langle m, k \rangle$ to denote that n is the pairing of numbers m ad k. 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 $(n, \psi(n))$, 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, if any, 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* well known fixed number,[4] 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 as before but with output 1 or 0. We define a *recursively enumerable* set S (r.e. S for short) as the collection of inputs

[2] A famous introduction to automated scientific discovery is the book by Pat Langley et al. [17].

[3] By *number* we refer to an element of the set of natural numbers (denoted by the symbol $\mathbb{N}$), unless otherwise specified.

[4] That depends on the very basic functions available in the language $\mathbb{X}$.

for which a program P generates output 1. If, besides this property, either P or another 1-equivalent $\mathbb{X}$-program P' [5] 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 in $\mathbb{X}$ for functions ψ 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 [13]).

First, the so-called s–1–1 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 $g(0)$, $g(1)$, $g(2)$, ... for $\psi(0, n)$, $\psi(1, n)$, $\psi(2, n)$, ..., respectively. Using the conventional 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 below.

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,

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

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, witnessing 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

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

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 j or less steps and $P(n)\downarrow_j i$ to say that P with input n outputs i in j or less steps.

It is difficult to suggest an empirical law in Physics that is not primitive recursive on the natural numbers. However, we will be working with the set of recursive functions that strictly includes the primitive recursive functions. Thinking about 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 [3].

3 The empiricist

Information for empiricists is provided in experimental notes, scientific articles, etc. E.g., original data for Galileo's experiment is given as a table for a function:

0	1	2	3	4	5	$\cdots$
$(0.1, 0.098)$	$(0.2, 0.392)$	$(0.3, 0.882)$	$(0.4, 1.568)$	$(0.5, 2.450)$	$(0.6, 3.528)$	$\cdots$

where the numbers in the top line provide some (arbitrary) order to the experimental data registered in the bottom line. In what follows we restrict numbers to the non-negative integers or natural numbers $\mathbb{N}$. As explained above, specific encoding techniques into the natural numbers for scientific laws expressed by means of rational/real numbers are discussed *inter alia* in [21].

A *text* T for a function is a map from numbers to pairs of numbers $(n, \text{value at } n)$. 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 $\psi \in \mathcal{R}$ in increasing order of the independent variable, as $\#\psi(0)\#\psi(1)\#\cdots\#\psi(t-1)\#$, where $\#$ is a separation symbol between numbers. [6] For each $\sigma \in SEG$, $content(\sigma)$ provides the corresponding set of pairs in σ and $content(T)$ provides the full set of pairs in the text T. Since σ is a prefix for a function, no such pairs (m, n_1) and (m, n_2), 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 $\widehat{\sigma}$, defined as

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

[6] Note that, in this case, the ordering number of the pairs is implicit:

0	1	2	3	4	$\cdots$
$(0, \psi(0))$	$(1, \psi(1))$	$(2, \psi(2))$	$(3, \psi(3))$	$(4, \psi(4))$	$\cdots$

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 Galileo did), she 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), she 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.

Although a parallel theory can be settled for sets, in this paper we focus mainly on functions as expressions of natural laws.

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

We give now an idea how an empirical law could be algorithmically established. Assume to start with that the scientist believes that all the empirical relations are primitive recursive. Then the universe of natural laws can be encoded into the natural numbers by means of standard encoding method. Let us assume for simplicity that the scientist is learning the number 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. [7] Let us see how the scientist discovers π...

Each time the suitable scientific method receives a new approximation of π, it outputs the same or a new conjecture, that is (the code of) a computer program that, for each number n, computes the nth-digit of the number being identified. 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 (the first $\mathbb{X}$-programme) 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. Input of further approximations, will probably make $\mathcal{M}$ change conjecture, one, twice, three times... until ...

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

```
Scientist M(#ψ(0)#ψ(1)#ψ(2)#···#ψ(n−1)# : SEG) : ℕ;
Var i, k ∈ ℕ;
Begin
    i := 0;
    k := 0;
    While k < n Do Begin
        If φ_i(k) = ψ(k) Then k ← k + 1
        Else Begin
            k ← 0;
            i ← i + 1
        End
    End;
    Conjecture i
End
```

Figure 1: Scientist $\mathcal{M}$ identifies the class of primitive recursive functions.

Definition 2 (Convergence of scientist, Gold [12]) *A scientist or scientific method $\mathcal{M}$ converges to a $\mathbb{X}$-program code p on text T 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 functions, Gold in [12]) *A scientist $\mathcal{M}$ Ex-*identifies *a 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.*

Identification of functions is called Ex-identification. Ex comes from **Ex**plain. Note that the concept of convergence is a limit concept. No one knows, at a particular time, if convergence is already achieved. Although this search for a code serves the purpose of learning all primitive recursive functions,[8] 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... It works with this method and it may work also with other methods of scientific discovery as well (see [17]), but that is another story. In what follows, Ex is the class of all Ex-identifiable sets of functions.

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:

Definition 4 *(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.*[9]

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

[9] 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 $t \in \mathbb{N}$, $\phi_e = \psi$.

To check that the class Ex is not empty consider two standard examples: The set AEZ of (total) computable functions ψ identical to 0 almost everywhere (**A**lmost **E**verywhere **Zero**) and the set SD of all (total) recursive functions ψ such that $\phi_{\psi(0)} = \psi$ (**S**elf **D**escribing). To see that the set AEZ is in Ex, just consider the scientist $\mathcal{M}$ that, on input $\sigma \in SEG$, constructs the canonical list μ of the pairs $(t, \psi(t))$ where the $\psi(t)$ is non-zero and outputs the code of the programme **Input** x**; If** $x \in dom(\widehat{\mu})$ **Then** $\widehat{\mu}(x)$ **Else** 0. As seen before in the last paragraphs of Section 2, the set SD is also Ex-identifiable by the scientist $\mathcal{M}$ that on input $\sigma \in SEG$ outputs 0 until $\psi(0)$ is seen and $\psi(0)$ afterwards.

4 Total methods

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

Definition 5 *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 such that (a) scientists can always output conjectures and (b) if a scientist can do with the values of the function ordered in increasing values of the independent variable, then other scientist can be devised to handled values of functions in any order.

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

Figure 2: Method $\mathcal{N}$ constructed from the possibly partial method $\mathcal{M}$.

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. $T[1] = T(0)$, $T[2] = T(0)T(1)$, $T[3] = T(0)T(1)T(2)$, ..., $T[t] = T(0)T(1)\cdots T(t-1)$, and $T[0]$ is the empty word. 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 was 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 2.

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, that is in the canonical form $\#\psi(0)\#\psi(1)\#\psi(2)\#\cdots$ (standing for $(0,\psi(0))$ $(1,\psi(1))$ $(2,\psi(2))$ ...), although any pair $(i,\psi(i))$ 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 proposition 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.

From this point on, *we will be consider only scientific methods that are total maps.* Moreover, without loss of generality, *scientific methods for functions will be operating only on canonical text.*

5 Rationality revisited

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 [2].

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 $T[0] = \#$, $T[1] = \#\psi(0)\#$, $T[2] = \#\psi(0)\#\psi(1)\#$, ..., $T[n] = \#\psi(0)\#\psi(1)\#\cdots\#\psi(n-1)\#$, 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 proven: 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.[10]

Consistency of a scientist $\mathcal{M}$ on a recursive function f can be expressed as $f[n] \subset \phi_{\mathcal{M}(f[n])}$. Let

$$[Ex]^{\text{consistent}} = \{S \subseteq Ex : S \text{ is } Ex\text{-identifiable by a consistent scientist}\} .$$

Proposition 3 $[Ex]^{consistent} \subset Ex$.

We choose the set SD to prove the separation between Ex and $[Ex]^{\text{consistent}}$. Obviously $SD \in Ex$. Given input $\sigma \in INIT$, the scientist conjectures $\widehat{\sigma}(0)$ and it is done; given any input $\sigma \in SEG$, the scientist conjectures 0 until reading $\widehat{\sigma}(0)$. However, $SD \notin [Ex]^{\text{consistent}}$. To see why not, consider the function f specified in Figure 3.

Function $f(y, x : \mathbb{N}) : \mathbb{N}$;
var $\sigma : INIT$; $i : \mathbb{N}$;
Begin
  $\sigma := (0, y)$;
  For $i := 1$ **To** x **Do**
    If $\mathcal{M}(\sigma\diamondsuit(i, 0)) \neq \mathcal{M}(\sigma)$ **Then** $\sigma := \sigma\diamondsuit(i, 0)$ **else** $\sigma := \sigma\diamondsuit(i, 1)$;
  Return $\widehat{\sigma}(x)$
End

Figure 3: $\mathcal{M}$ is any total scientist.

Now we apply Kleene's Theorem to write $\phi_e(x) = f(e, x)$, with $\phi_e \in SD$. Note that $\lim \sigma = T$ is a text for ϕ_e. The interesting point is that once we assume that scientist $\mathcal{M}$ for SD is consistent, we get that either $\mathcal{M}(\sigma\diamondsuit(i, 0)) \neq \mathcal{M}(\sigma)$ or $\mathcal{M}(\sigma\diamondsuit(i, 1)) \neq \mathcal{M}(\sigma)$, and, consequently, for every $x \in \mathbb{N}$, $\mathcal{M}(\phi_e[x + 1] \neq \mathcal{M}(\phi_e[x])$ meaning that the scientist $\mathcal{M}$ changes her mind infinitely many times not converging to any code of ϕ_e, not Ex-identifying ϕ_e.

The previous result may appear as a consequence of the effort of the scientist to keep consistency in every data for every recursive function, even for those functions that are not her expertise. However, a strict concept of consistent scientist may be introduced, the concept of class-consistent scientist. A scientist $\mathcal{M}$ Ex-identifying a set S of recursive functions is said to be class-consistent if she is consistent in every function belonging to S, but not necessarily consistent

[10] 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.

in the functions not belonging to S. This definition sounds like a scientist being consistent in her own expertise. However, Janis Bārzdiņš proved in [1], that class-consistent scientists Ex-identify a strict subset of the Ex-identifiable sets. I.e. such a scientist, or a method, have less identification power than those that may standby in inconsistency during the learning phase, that is, before they syntactically converge to the final hypothesis.

6 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 3, 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 Galileo'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 imperative $\mathbb{X}$-routines 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 [18]. As a consequence, the primitive recursive functions can be enumerated and recursively encoded into the natural numbers.

All primitive recursive functions are Ex-identifiable in the sense of Definitions 1, 2, and 3 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 $0, 1, 2, \ldots$ and providing as output the first conjecture consistent with the input data/measurements done thus far (see Section 3). 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, we may conclude 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. In general, subclasses of the class $\mathcal{R}$ of recursive functions cannot be identified in the limit by single general routines (see [15]). The paradigm of induction (or Ex-identification) so far described can be synthesised in the this way:

CRITERION 1: CONVENTIONAL SCIENTISTS : A scientist *converges syntactically* to a law, that is the scientist converge to a number $p \in \mathbb{N}$ such that ϕ_p is the correct law.

7 Function identification with anomalies

Although we can Ex-identify sets of infinitely many recursive functions, the Ex paradigm is not able to capture the full set $\mathcal{R}$ of recursive functions by means of a single scientist or a single method. Possibly we can proceed in a different way. Since a recursive function is an infinite object, surely we agree that to identify the function in all but finitely many points is better than not identify the function at all. By weakening identification criteria we might be able to identify larger collections of sets of recursive functions, namely $\mathcal{R}$ itself.

Definition 6 *(n-variant) We say that a partial recursive function ξ is an n-variant of a function $f \in \mathcal{R}$, if ξ coincides with f in all but finitely many points in number not exceeding n (and we write $f =^n \xi$).*

Definition 7 *($\star$-variant) We say that a partial recursive function ξ is an $\star$-variant of a function $f \in \mathcal{R}$, if ξ coincides with f in all but finitely many points (and we write $f =^\star \xi$).*

According to Robert Daley [11, 15], if a function ξ is a n-variant or a $\star$-variant of a function $f \in \mathcal{R}$, then, for each of the inputs where ξ differs from f, either ξ is not defined, and it is called an error of omission, or it is defined but its value differ from the corresponding value of f, and it is then called an error of commission. E.g. finding a lack of agreement in a predicted trajectory of a planet is an error of commission; not providing any answer at all (in finite time) for the position of the planet at instant t is an error of omission.

Definition 8 *(Ex^n-identification, Case and Smith [9, 10]) A computable scientist Ex^n-identifies a function $\psi \in \mathcal{R}$, if there exists an order $p \in \mathbb{N}$ such that, for every $t \geq p$, $\mathcal{M}$ on input $\psi[t]$ conjectures the same code of an n-variant of ψ. We say that the scientist $\mathcal{M}$ Ex^n-identifies $S \in \mathcal{R}$ if Ex^n-identifies every function $\psi \in S$.*

A recursive function or a set of recursive functions is said to be Ex^n-identifiable if there exists a scientist $\mathcal{M}$ that Ex^n-identifies that function or set of functions (respectively).

Definition 9 *($Ex^\star$-identification, Blum and Blum [2]) A set $S \in \mathcal{R}$ is said to be of class $Ex^\star$ if every function $\psi \in S$ is in Ex^n for some $n \in \mathbb{N}$.*

The former definition is equivalent to say that a computable scientist $Ex^\star$-identifies the function $\psi \in \mathcal{R}$, if there exists an order $p \in \mathbb{N}$ such that, for every $t \geq p$, $\mathcal{M}$ on input $\psi[t]$ conjectures the same code of a $\star$-variant of ψ. We say that the scientist $\mathcal{M}$ $Ex^\star$-identifies $S \in \mathcal{R}$ if $Ex^\star$-identifies every function $\psi \in S$.

Definition 10 *Ex, Ex^n, and $Ex^\star$, are the corresponding classes of sets which are Ex-, Ex^n-, or $Ex^\star$-identifiable by computable scientists.*

The class Ex^0 will be denoted also by Ex.

8 The Ex hierarchy

The first task is to identify possible separations between the classes Ex^m and Ex^n, for $m \neq n$, and between classes Ex^n and $Ex^\star$, for every n. For that purpose we prove that there are non-Ex^n-identifiable recursive functions in each new level $n+1$.

Definition 11 *ASD^n, for $n \in \mathbb{N}$, is the set of all recursive functions $\psi \in \mathcal{R}$ such that $\psi(0)$ is an index for an n-variant of ψ, that is $\phi_{\psi(0)} =^n \psi$. $ASD^\star$ is the set of all recursive functions $\psi \in \mathcal{R}$ such that $\psi(0)$ is an index for an $\star$-variant of ψ, that is $\phi_{\psi(0)} =^\star \psi$.*

The class ASD^0 is just the same class SD. The class ASD^n is Ex^n-identifiable by a straightforward method: the scientist waits until the input is long enough to include the value of the function at 0, i.e. the pair $(0, \psi(0))$, and then she outputs the value $\psi(0)$.

The proof of the separation $Ex^n \subset Ex^{n+1}$ is based in the following idea. Considering an arbitrary scientist $\mathcal{M}$ that presumably Ex^n-identify ASD^{n+1}, we extend (in a function compatible way) a given prefix σ by suitable segments $\tau \in SEG$ that make the scientist $\mathcal{M}$ changing her mind, i.e. $\mathcal{M}(\sigma \diamondsuit \tau) \neq \mathcal{M}(\sigma)$ (followed by $\sigma := \sigma \diamondsuit \tau$). If this process succeeds, Kleene's Theorem guaranties that the resulting $\lim \sigma$ is a text for a function in ASD^{n+1} that the scientist fails to Ex^n-identify. If the process cannot be continued just because from some time on $\mathcal{M}$ does not change its mind, then the construction makes sure that $\mathcal{M}$ does not distinguish between $n+1$ different functions in ASD^{n+1}. To obtain all these ASD^{n+1} different functions, Kleene's Theorem is applied to a single partial function undefined in n contiguous points.

Proposition 4 *(Case e Smith [9, 10], Jain et al. [15]) For every $n \in \mathbb{N}$, $ASD^{n+1} \in (Ex^{n+1} - Ex^n)$.*

Function $f(e, x : \mathbb{N}) : \mathbb{N}$;
Const: $n \in \mathbb{N}$;
Var: $k, \ell, y : \mathbb{N}$; D : **set of** $\mathbb{N}$;
Begin
$\sigma := (0, e)$; % σ stands for a finite approximation of f
$\ell := 1$; % Number of points in the domain of $\widehat{\sigma}$
$D := \{0\}$; % Domain of $\widehat{\sigma}$
While[1] *true* **Do Begin**
For $y := 0$ **To** $n + 1$ **Do** % The $n + 2$ possible continuations of σ
$\tau_y := \sigma \diamondsuit (\ell, y) \diamondsuit \ldots \diamondsuit (\ell + n, y)$;
$k := \ell + n$;
While[2] *true* **Do Begin**
$k := k + 1$
For $y := 0$ **To** $n + 1$ **Do Begin**
$\tau_y := \tau_y \diamondsuit (k, 0)$;
If $\mathcal{M}(\sigma) \neq \mathcal{M}(\tau_y)$ † **Then Exit[2]**
End;
$D := D \cup \{k\}$;
If $x \in D$ **Then Return** $\widehat{\tau}_0(x)$;
End[2];
$\sigma := \tau_y$;
$D := D \cup \{\ell, \ell + 1, \ldots, \ell + n\}$;
If $x \in D$ **Then Return** $\widehat{\sigma}(x)$;
$\ell := k + 1$
End[1]
End

Figure 4: Specification of a function $f \in ASD^{n+1}$ that is not Ex^n-identifiable.

We define a function of ASD^{n+1} that escapes to Ex^n-identification. Suppose that some total computable scientist $\mathcal{M}$ Ex^n-identifies ASD^{n+1} and consider the (possibly partial) binary function f specified in Figure 4. For a fixed value of e, if we list the stream of values of f as a function of x, then we get an infinite sequence with prefixes of the form:

$$e \underbrace{a_1 \ldots a_1}_{n+1} \overbrace{0 \ldots 0}^{n_1} \underbrace{a_2 \ldots a_2}_{n+1} \overbrace{0 \ldots 0}^{n_2} \ldots \tag{1}$$

possibly ending with an infinite sequence of 0s, where the number of intermixed 0's may vary from segment to segment. The values of the a's are within $n + 2$ possible values from 0 to $n+1$. There are two cases to consider: (a) the internal While[2] loop always terminates: function f obeys to the pattern (1); (b) from some order on, the internal loop does not halt: in this case, the function f obeys to the pattern:

$$e \underbrace{a_1 \ldots a_1}_{n+1} \overbrace{0 \ldots 0}^{n_1} \ldots \underbrace{a_k \ldots a_k}_{n+1} \overbrace{0 \ldots 0}^{n_k} \underbrace{\perp \ldots \perp}_{n+1} 0^\omega \tag{2}$$

i.e., function f is undefined in a segment of $n + 1$ input values followed by an infinite number of 0s.

Relatively to Figure 4, let e be a number provided by Kleene's second recursion theorem such that $\phi_e(x) = f(e, x)$.

In case (a), given the text $\lim \sigma$ generated for the function f, the scientist $\mathcal{M}$ is forced to change her mind infinitely often (as in Exit[2]), so that $\mathcal{M}$ does not Ex^n-identify $f \in SD \subset ASD^{n+1}$.

In case (b), i.e. whenever, for some value of x, the internal loop does not terminate, the function f is undefined in $n+1$ contiguous points. Inter alia, we list $n+2$ functions that the scientist $\mathcal{M}$ cannot fully distinguish as can easily be seen in the middle step marked by a † in Figure 4:

$$
\begin{aligned}
f_0 &= e\underbrace{a_1 \ldots a_1}_{n+1}\overbrace{0\ldots0}^{n_1}\ldots\underbrace{a_k \ldots a_k}_{n+1}\overbrace{0\ldots0}^{n_k}\underbrace{\boxed{0}\ldots\boxed{0}}_{n+1}\,0^\omega \\
f_1 &= e\underbrace{a_1 \ldots a_1}_{n+1}\overbrace{0\ldots0}^{n_1}\ldots\underbrace{a_k \ldots a_k}_{n+1}\overbrace{0\ldots0}^{n_k}\underbrace{\boxed{1}\ldots\boxed{1}}_{n+1}\,0^\omega \\
&\vdots \\
f_{n+1} &= e\underbrace{a_1 \ldots a_1}_{n+1}\overbrace{0\ldots0}^{n_1}\ldots\underbrace{a_k \ldots a_k}_{n+1}\overbrace{0\ldots0}^{n_k}\underbrace{\boxed{n+1}\ldots\boxed{n+1}}_{n+1}\,0^\omega
\end{aligned}
$$

Therefore, with a permutation $b_0 b_1 b_2 \cdots b_n$ of $n+1$ different numbers from 0, 1, ..., $n+1$, filling the $n+1$ undefined cells of f, the scientist $\mathcal{M}$ can only Ex^n-identify $n+1$ n-variants of the functions f_0, f_1, ..., f_{n+1} above, leaving at least one to identify. Consider

$$
f'(x) = \begin{cases} y' & \text{se } x = \ell, \ldots, \ell + n \\ f(x) & \text{otherwise} \end{cases} ,
$$

where $y' \in \{0, \ldots, n+1\}$ is the value that the sequence $b_0 b_1 b_2 \cdots b_n$ cannot cover. The scientist $\mathcal{M}$ does not Ex^n-identify $f' \in ASD^{n+1}$. Since $\mathcal{M}$ is an arbitrary scientist, we conclude that $ASD^{n+1} \notin Ex^n$. Now, it is straightforward to conclude that $ASD^\star \in Ex^\star$. Let us suppose that $ASD^\star \in \cup_{n\in\mathbb{N}}\ Ex^n$. Then, there exists a k such that $ASD^\star \in Ex^k$. In particular, we have that $ASD^{k+1} \in Ex^k$ which contradicts Proposition 4. We have then the following proposition:

Proposition 5 (Case and Smith [9, 10]) $ASD^\star \in (Ex^\star - \cup_{n\in\mathbb{N}}\ Ex^n)$.

It follows that accepting anomalies increases the power of scientific inference, in an infinite non-collapsing hierarchy of collections of classes of sets:

Proposition 6 $Ex = Ex^0 \subset Ex^1 \subset Ex^2 \subset \cdots \subset Ex^n \subset \cdots \subset Ex^\star$.

If a function ξ is a n-variant or a $\star$-variant of a function $f \in \mathcal{R}$, then, for each of the inputs where ξ differs from f, either ξ is not defined (error of omission) or it is defined but its value differ from the corresponding value of f (error of commission). Tolerating anomalies means that the scientist can forecast the system up to a finite number of input values for which the conjecture does not provide the right answer, either by diverging (error of omission) or not converging appropriately (error of commission). Errors of commission can be corrected and correspond to the points where the conjecture differs from the input tape. The new conjecture that results from the elimination of commission errors is such like a general law with exceptions. In [8] (page 7), talking about anomalous index of refraction of X-rays, John Case explains that *a few anomalies being tolerated in final predictive explanations, came from anomalous dispersion: the classical explanation for the degree of bending of "light" passing through a prism, fails for the X-ray case, an anomalous case.*

Since the general law holds for infinitely many points, a scientist that tolerates a finite number of errors can identify countably many more recursive functions than a scientist that does not tolerate omissions. Note that the supplement of functions that such a scientist can identify are not those partial functions she conjectures, but recursive functions from $\mathcal{R}$. The scientist identify them up to a finite number of errors.

John Case writes in [8] *Hence, tolerating anomalies strictly increases the inferring power [...] The anomalies that must be exploited to prove the Ex^n-hierarchy above are anomalies of omission or incompleteness: the predictive explanation's errors are where they loop infinitely with no prediction (see [9, 10]) [...] Hence, thanks to the unsolvability of the Halting Problem ([13]), Popper's Refutability Principle ([20]) is violated in a way Popper did not consider (see [9, 10])!*

Obviously, such a paradigm transgresses Popper's strict doctrine. In fact, we know that: $ASD^1 \in (Ex^1 - Ex)$. However, a scientist $\mathcal{M}$ that converges to the hypothesis e, might not be supporting Popper's refutability, for the simple reason that ϕ_e might not be defined on some input y. Hypothesis e may not be falsified: (a) it is not known if the instance ϕ_e is undefined on some input y, and (b) if so, programme $\{e\}$ on input y does not halt, so that one cannot prepare any experimental apparatus to refute "theory $\mathcal{T}$ on y", given a basic statement such as $\phi_e(y) \neq \psi(y)$, since it is not even known with generality if $\{e\}(y)$ halts or not and, consequently, produce a prediction refutable by observation.

Thus, another paradigm emerges that extends strict Ex-identification to a world of scientific laws valid in all but finitely many cases:

CRITERION 2: ALMOST CONVENTIONAL SCIENTISTS : A scientist *converges syntactically* to a law with exceptions, that is the scientist converges to a number $p \in \mathbb{N}$ such that ϕ_p is the correct law up to finitely many errors or exceptions.

A scientist that permits a finite number of errors can state a law that allows the simulation of the phenomenon for all but finitely many values of the variables.

We question now whether $Ex^\star$ includes $\mathcal{R}$...

9 Behaviourally correct identification

The proof that $\mathcal{R}$ is not in $Ex^\star$ and therefore $Ex^\star$ does not exhaustively cover $\mathcal{R}$ is done by relaxing the restriction of convergence of scientists to single programmes. Although it leads to the definition of more liberal paradigms of identification, the fact is that an adequate paradigm of identification was already defined by the time this proof was done:

Definition 12 *(Bc-identification, Bārzdiņš [1], Case and Smith [9, 10]) A computable scientist $\mathcal{M}$ Bc-identifies the recursive function $\psi \in \mathcal{R}$ if there exists an order $p \in \mathbb{N}$ such that, for every $n \geq p$, $\phi_{\mathcal{M}(\psi[n])} = \psi$. We say that the scientist $\mathcal{M}$ Bc-identifies a set of recursive functions $S \subseteq \mathcal{R}$ if, for every $\psi \in S$, $\mathcal{M}$ Bc-identifies ψ.*

The letters in Bc stand for **B**ehaviourally **C**orrect.

Definition 13 *(Bc^n-identification, Case and Smith [9, 10]) A computable scientist $\mathcal{M}$ Bc^n-identifies the recursive function $\psi \in \mathcal{R}$ if there exists an order $p \in \mathbb{N}$ such that, for every $t \geq p$, $\mathcal{M}(\psi[t])$ is a code of a n-variant ξ of the function ψ. We say that the scientist $\mathcal{M}$ Bc^n-identifies a set of recursive functions $S \subseteq \mathcal{R}$ if, for every $\psi \in S$, $\mathcal{M}$ Bc^n-identifies ψ.*

Definition 14 *($Bc^\star$-identification, Case and Smith [9, 10]) A computable scientist $\mathcal{M}$ $Bc^\star$-identifies the recursive function $\psi \in \mathcal{R}$ if there exists an order $p \in \mathbb{N}$ such that, for every $t \geq p$, $\mathcal{M}(\psi[t])$ is a code of a $\star$-variant ξ of the function ψ. We say that the scientist $\mathcal{M}$ $Bc^\star$-identifies a set of recursive functions $S \subseteq \mathcal{R}$ if, for every $\psi \in S$, $\mathcal{M}$ $Bc^\star$-identifies ψ.*

The following statement sounds intriguing at first sight but it is straightforward to prove. Essentially, it says that all the recursive functions thus far identified by (Ex-, Ex^n-, and $Ex^\star$-) scientists (converging syntactically) can be identified with no errors (!) by the new scientists converging semantically. Such a scientist converge to a single function but not to single programming code.

Let $S \in Ex^\star$, witnessed by scientist $\mathcal{M}$, and $\psi \in S$. From scientist $\mathcal{M}$, we build a new scientist $\mathcal{M}'$ which behaves as follows: $\mathcal{M}'$ simulates $\mathcal{M}$ on its own input

$$\sigma[t] = (0, \sigma(0))(1, \sigma(1)) \dots (t-1, \sigma(t-1)) \ ,$$

getting code $e = \mathcal{M}(\sigma[t])$, and builds from the same input a looking up ordered list μ of the different pairs $(i, \sigma(i))$ occurring in $\sigma[t]$; with that index e and such a list μ, the scientist $\mathcal{M}'$ outputs the code of the programme of Figure 5.

Of course, the code of such a programme is always changing (since the constant μ is changing), i.e. scientist $\mathcal{M}$ changes her mind infinitely often but, from some order on, namely passed all anomalies of ϕ_e, the codes of the functions χ are all different codes of the same ψ!

```
Function χ(x : ℕ) : boolean;
    Const μ : list of ℕ × ℕ;
Begin
    Case x of
        x ∈ dom(μ̂) :      Return μ̂(x);
        x ∉ dom(μ̂) :      Return {M(μ)}(x)
    End
End
```

Figure 5: Scientist that witnesses the inclusion $Ex^n \subseteq Bc$.

We then conclude that

Proposition 7 *(J. Steel cited by Case and Smith [10]) $Ex^\star \subseteq Bc$.*

The proof that, for each n, Ex^n is strictly included in Bc can also be done in a different way. Let $S \in Ex^n$, for some fixed n, be witnessed by scientist $\mathcal{M}$, and let $\psi \in S$. From scientist $\mathcal{M}$, we build a new scientist $\mathcal{M}'$ which behaves as follows: Whenever σ contains more than n distinct points of ψ, $\mathcal{M}'$ simulates $\mathcal{M}$ on its own input $\sigma[t]$, getting code $e = \mathcal{M}(\sigma[t])$; then $\mathcal{M}'$ runs in parallel the program $\{e\}$ on every single non repeated item of σ and halts the computation whenever less than $n+1$ runs did not halt or converge to the wrong value; finally, $\mathcal{M}'$ corrects e on all these points and outputs a modified code e'. In all the other cases, $\mathcal{M}'$ is left undefined.

The scientists that are able to rewrite correct codes from conjecture to conjecture can do what scientists converging syntactically to hypotheses with finitely many errors can and more, without errors:

CRITERION 3: UNCONVENTIONAL SCIENTISTS : A scientist *converges semantically* to a law (with no exceptions), that is, from some order k on, the scientist conjectures numbers $p_k, p_{k+1}, \ldots$, such that, for all $i \geq k$, ϕ_{p_i} is a correct version of the law.

10 Separation results for $Ex^\star \subseteq Bc$

The set of functions chosen to help in such a separation are introduced in the next definition:

Definition 15 *$\mathcal{S}$ is the set of all recursive functions $\psi \in \mathcal{R}$ such that, for all but finitely many i, $\psi(i)$ is an index of ψ, that is, for all but finitely many i, $\phi_{\psi(i)} = \psi$.*

We first argue that set $\mathcal{S}$ is not empty. The proof is straightforward. Let $f : \mathbb{N}^2 \to \mathbb{N}$ be defined as $f(\alpha, x) = \alpha$. Applying the $s-1-1$ theorem to f, we write $\phi_{s(\alpha)}(x) = \alpha$ for some primitive recursive function s. Then Kleene's theorem provides a computable e such that $\phi_e(x) = e$. The function ϕ_e takes value e everywhere and belongs to $\mathcal{S}$. Since there are infinitely many such indexes, the set $\mathcal{S}$ is countably infinite.

Proposition 8 $\mathcal{S} \in Bc$.

For the identification of $\mathcal{S} \in Bc$, the scientist just have to output the last value of the function seen thus far in the input.

Let $\mathcal{F}$ denote the class of all partial functions with signature $\mathbb{N} \to \mathbb{N}$. An operator is a total function $\Phi : \mathcal{F} \to \mathcal{F}$.

We describe a way in which an operator can be defined by means of a computer program P. The program P has access to its input, a text T for a function $f : \mathbb{N} \to \mathbb{N}$ and a value n, to compute $\Phi(f)(n)$. The function f can be total or partial. Let us assume that the computer program P operates in a way such that, at any moment of the computation of P over f and n, only a finite number of function pairs of T are available. We then say that the operator is recursive and computed by the program P. If the program P searches the input stream T not finding the required pair $(i, f(i))$, for some natural number i, of the graph of f, then this search runs forever since the text for a function is always infinite in extension, either by means of the repeated separation symbol #, or because the values of the function are repeated infinitely many times. In the proofs that follows, we will make extensive use of this concept. Note that an operator can be defined over non-computable functions, but to be recursive it has to algorithmically transform the prefixes of the input function f into the prefixes of the output function $\Phi(f)$, since at any time only a finite prefix of f, let us say of some size k, is available to compute a prefix $(0, \Phi(f)(0))$ $(1, \Phi(f)(1))$ $(2, \Phi(f)(2))$ ... $(n-1, \Phi(f)(n-1))$ of size n.

The most relevant mathematical result on recursive operators for what to follows is John Cases's theorem:

Proposition 9 *If Ξ is a recursive operator, then there is a recursive, monotone increasing function $h : \mathbb{N} \to \mathbb{N}$ such that, for all $n, x \in \mathbb{N}$, $\phi_{h(n)}(x) = \Xi(h)(\langle n, x\rangle)$, where $\langle n, x\rangle$ is the pairing or joint encoding of the values n and x.*

We can now prove a separation between classes $Ex^\star$ and Bc, so that Bc criterion has more identification power than $Ex^\star$ criterion. As a consequence, we have that neither Ex^n, in any level, nor $Ex^\star$ include the full set $\mathcal{R}$ of recursive functions.

We first discuss the idea of the proof. We intend to construct from successive prefixes $\sigma_0[0], \sigma_0[1] \cdots \sigma_0[n]$, a stepwise extension of some prefix $\sigma_0 \in SEG$, a total function $\lim \widehat{\sigma_0} \in \mathcal{S}$. This extension is done in a way such that the "$Ex^\star$-"scientist $\mathcal{M}$ is compelled to change her mind infinitely many times, thus failing to $Ex^\star$-identify the corresponding function $\lim \widehat{\sigma}_0$ ($\in \mathcal{S}$). To perform the intended extension, we consider two other auxiliary extensible prefixes σ_1 and σ_2 that, in the limit, may also become graphs of functions in $\mathcal{S}$. While $\mathcal{M}$ does not distinguish σ_0 from σ_1 or σ_0 from σ_2, the prefixes σ_1 and σ_2 are pointwise extended. If no distinction is found, then $\mathcal{M}$, a scientist that supposedly is capable to $Ex^\star$-identify functions in $\mathcal{S}$ (up to finitely many errors), fails to distinguish between σ_1 and σ_2 although in the limit the graphs of $\widehat{\sigma}_1$ and $\widehat{\sigma}_2$ differ in infinitely many points. In the other way round, the scientist $\mathcal{M}$ conjectures

differently on σ_1 and σ_0 or on σ_2 and σ_0. In this case σ_0 is extended in a way to follow either σ_1 or σ_2 but making the scientist to change her mind on the extended text of σ_0. From a stage m on, σ_1 and σ_2 are forced to follow σ_0 in their values. The process continues, new prefixes in SEG being defined, σ_0 being continued as one of the new σ_{2m+1} or σ_{2m+2}, and the former σ_k, from $k = 1$ to $k = 2m$, extended to follow σ_0. If the extension procedure fails, we get the same situation as discussed in the beginning, with the scientist being unable to distinguish between two different functions in $\mathcal{S}$, σ_{2m+1} or σ_{2m+2} for some m. If the process can run forever, then the function with graph $\lim \sigma_0$ has values that are their own codes by application of theorem 9. Such a function with graph $\lim \sigma_0$ is not $Ex^\star$-identifiable by $\mathcal{M}$ and the proof is done.

Proposition 10 *(Case and Smith [9, 10], Jain et al. [15]) $Ex^\star \subset Bc$.*

We now discuss the proof in full detail. Let $\mathcal{M}$ be any (total) scientist. With the help of the functional specified in Figure 6 and Proposition 9, we show that the scientist $\mathcal{M}$ cannot $Ex^\star$-identify some functions in $\mathcal{S}$. Considering the specification of Figure 6, in each execution of the "While" loop the program attempts to define two new functions that in the limit and considering Proposition 9 are in $\mathcal{S}$. According with Proposition 9, there exists a total increasing (primitive) recursive function h such that $\Xi(h)(\langle k, x\rangle) = \phi_{h(k)}(x)$. The following are the main characteristics of the specification applied to such function h:

Functional $\Xi(h : \mathbb{N} \to \mathbb{N}; \langle k, x\rangle : \mathbb{N}) : \mathbb{N}$;
Var $\sigma : \mathbb{N} \to SEG$; $m, y, s : \mathbb{N}$; % Notation h_n, σ_n;
Begin
  $y := 0$;
  $\sigma_0 := (0, h_0)$;
  If $\langle k, x\rangle = 0$ **Then Return** h_0;
  For $m := 0$ **To** $+\infty$ **Do Begin**
    $\sigma_{2m+1} := \sigma_0$;
    $\sigma_{2m+2} := \sigma_0$;
    While $\mathcal{M}(\sigma_{2m+1}) = \mathcal{M}(\sigma_0)$ **And** $\mathcal{M}(\sigma_{2m+2}) = \mathcal{M}(\sigma_0)$ **Do Begin**
      $y := y + 1$;
      $\sigma_{2m+1} := \sigma_{2m+1}\diamondsuit(y, h_{2m+1})$;
      $\sigma_{2m+2} := \sigma_{2m+2}\diamondsuit(y, h_{2m+2})$;
      If $k \in \{2m + 1, 2m + 2\}$ **And** $\langle k, x\rangle = y$ **Then Return** h_k
    End While;
    If $\mathcal{M}(\sigma_{2m+1}) \neq \mathcal{M}(\sigma_0)$ **Then** $\sigma_0 := \sigma_{2m+1}$ **Else** $\sigma_0 := \sigma_{2m+2}$;
    For $s := 1$ **To** $2m$ **Do** $\sigma_s := \sigma_s\diamondsuit(|\sigma_s|, \widehat{\sigma}_0(|\sigma_s|))\diamondsuit \cdots \diamondsuit(y, \widehat{\sigma}_0(y))$;
    If $k \leq 2m$ **And** $\langle k, x\rangle \leq y$ **Then Return** $\widehat{\sigma}_k(\langle k, x\rangle)$
  End For
End

Figure 6: Recursive Ξ functional.

1. The sequences σ_0, σ_1, ..., σ_{2m}, σ_{2m+1}, σ_{2m+2} of SEG are all prefixes of potential graphs of total functions of $\mathcal{S}$. In each step m of the external "For" loop the two graph prefixes σ_{2m+1} and σ_{2m+2} are created, and whenever the "While" loop halts, the domain of each of the functions $\widehat{\sigma}_k$, from $k = 1$ to $k = 2m$, is updated in the internal "For" loop, so that they follow the values of $\widehat{\sigma}_0$.

2. In the case that, at some final step m of the main "For" loop, the "While" does not terminate, the prefixes σ_0, σ_1, ..., σ_{2m} stand as graphs of partial functions, but the functions $\lim \widehat{\sigma}_{2m+1}$ and $\lim \widehat{\sigma}_{2m+2}$ are totally defined. The scientist $\mathcal{M}$ cannot $Ex^\star$-identify $\mathcal{S}$ since she is not able to distinguish in the limit between $\phi_{h(2m+1)}$ and $\phi_{h(2m+2)}$. According to Proposition 9 and the specified functional, we have that $\phi_{h(2m+1)}(x) = h(2m+1)$ and $\phi_{h(2m+2)}(x) = h(2m+2)$, for every $x \geq x_0$, for some order x_0:

$$\mathcal{S}^{n+1} \supset \mathcal{S} \ni \phi_{h_{2m+1}} \quad = \quad h_0 \quad \underbrace{h_? \cdots h_?}_{\text{mix of even and odd indexes}} \quad h_{2m+1} h_{2m+1} \cdots h_{2m+1} \cdots$$

$$\mathcal{S}^{n+1} \supset \mathcal{S} \ni \phi_{h_{2m+2}} \quad = \quad h_0 \quad \underbrace{h_? \cdots h_?}_{\text{mix of even and odd indexes}} \quad h_{2m+2} h_{2m+2} \cdots h_{2m+2} \cdots$$

   These two functions are different since h is an increasing function by the same Proposition 9, but the scientist $\mathcal{M}$ can only permit a finite number of errors in $Ex^\star$-identification. Note that, although h is total, $\Xi(h)(\langle k, x\rangle)$ is undefined for every $k > 2m+2$, meaning that h_k is a code of the everywhere undefined function, for every $k > 2m+2$.

3. If the "While" loop halts for every m, then $\lim \widehat{\sigma}_k$ is a total function, for every $k \in \mathbb{N}$. In this case, all the values of the total increasing recursive function h, such that $\phi_{h_k}(x) = \Xi(h)(\langle k, x\rangle)$, for every $k \in \mathbb{N}$, are codes of h itself. In this case, the scientist $\mathcal{M}$ cannot $Ex^\star$-identify $\mathcal{S}$ since she does not converge on the text $\lim \sigma_0$ as the guard of the "While" loop indicates. Taking numbers $i, j \in \mathbb{N}$, such that $i \neq j$, either ϕ_{h_i} and ϕ_{h_j} coincide in all points, or ϕ_{h_i} and ϕ_{h_j} differ only in a finite number of points. The first case is due to the fact that σ_0 follows σ_k, for some k even or odd, until some order ℓ, and then σ_k follows σ_0. The second case is due to the fact that σ_0 follows some σ_k, let us say for odd k, until some point x_0, and then both σ_k and σ_{k+1} follows σ_0, but then both σ_k and σ_{k+1} will be differing in finitely many points. Thus, for every k, h_k is an index of h in all but finitely many points k.

To sum up, the two cases to be considered are the following:

1. The "While" loop always terminate: $\phi_{h_k} \in \mathcal{S}$, for every $k \in \mathbb{N}$. Since $\mathcal{M}$ should converge on $\phi_{h_0}[y]$ for sufficient large y, the scientist should change her mind only finitely many times; however that is not the case due to the guard of the "While" clause and the updates of the following "If" clauses.

2. The "While" loop does not terminate for some m: in this case, we take the functions $\phi_{h_{2m+1}}$ and $\phi_{h_{2m+2}}$, both in $\mathcal{S}$. However, we have that $\mathcal{M}(\sigma_{2m+1}) = \mathcal{M}(\sigma_0) = \mathcal{M}(\sigma_{2m+2})$, meaning that in the limit the scientist $\mathcal{M}$ does not distinguish between $\phi_{h_{2m+1}}$ and $\phi_{h_{2m+2}}$ both in $\mathcal{S}$.

As a consequence of the previous result we have that

Proposition 11 $\mathcal{R} \notin Ex^\star$.

This last statement holds since $Ex^\star$ does not contain all of the recursive functions in $\mathcal{S}$ and consequently cannot contain $\mathcal{R}$.

In our view the best example of Bc-identification is the case of learning neural networks, such like the learning of a function according with Kolmogorov's Theorem (see [14]). Neural nets are made of interconnected logic gates. Each connection between two units is weighted. In the most general case, the "program" is the table of interconnections and weights of dimension $n \times n$ for n computation logic units. Weight zero means no connection between the respective units. In the learning phase, that can endure forever, the weights can be recomputed each time some new patterns/observations are classified. After some time, this process induces only slight changes in the weights. Despite the changes, the net is able to stabilise in correct classifications. In the common case, some patterns are misclassified and can be considered exceptions, so that, in the next section, we will consider Bc-identification with errors. Surprisingly, this is one way to identify the full set $\mathcal{R}$ of recursive functions...

11 The Bc hierarchy

The set of functions chosen to help in the separations that follow is a slight modification of $\mathcal{S}$:

Definition 16 *The set $\mathcal{S}^n$, for $n \in \mathbb{N}$, is the set of all recursive functions $\psi \in \mathcal{R}$ such that, for all but finitely many $i \in \mathbb{N}$, $\psi(i)$ is an index for an n-variant of ψ, that is, for all but finitely many i, $\phi_{\psi(i)} =^n \psi$. $\mathcal{S}^\star$ is the set of all recursive functions $\psi \in \mathcal{R}$ such that, for all but finitely many $i \in \mathbb{N}$, $\psi(i)$ is an index for an $\star$-variant of ψ, that is, for all but finitely many $i \in \mathbb{N}$, $\phi_{\psi(i)} =^\star \psi$.*

Proposition 12 *For all $n \in \mathbb{N}$, $\mathcal{S}^{n+1} \in Bc^{n+1}$.*

The scientist outputs the last value of the function seen so far in the input.

Proposition 13 *(Case and Smith [9, 10], Jain et al. [15]) For each $n \in \mathbb{N}$, $\mathcal{S}^{n+1} \in (Bc^{n+1} - Bc^n)$.*

Again, we specify a recursive procedure that successively defines prefixes of graphs of functions, σ_0, σ_1, ..., σ_n, ... Starting from σ_0, the program searches for $m+1$ values where the conjecture $\mathcal{M}(\sigma_0)$ is defined. Then σ_0 is extended to σ_1 that differs from σ_0 in the same $m+1$ values thus found. At stage n, starting from σ_n, the program searches for $m+1$ values where the conjecture $\mathcal{M}(\sigma_n)$ is defined. Then σ_0 is extended to σ_n that differs from σ_0 in the same $m+1$ values. In all stages the other σ_i, for $i < n$ follow σ_0: σ_0, ..., σ_{n-1} have been defined in some specified domain; then σ_0 follows σ_n up to $m+1$ values where σ_0 and σ_n differ; from that point on, σ_1, ..., σ_n follows σ_0. For $i, j \in \mathbb{N}$, $i \neq j$, in the limit σ_i differs from σ_j in $m+1$ values. Then, in the limit $\lim \sigma_0$ is a text for a function h such that, for every $k \in \mathbb{N}$, ϕ_{h_k} has values that are codes of h itself up to $m+1$ errors.

Functional $\Xi(h : \mathbb{N} \to \mathbb{N}; \langle k, x\rangle : \mathbb{N}) : \mathbb{N}$;
Var $\sigma : \mathbb{N} \to SEG$; $i, j, q, y, y', z, s, \ell_0, \ldots \ell_m : \mathbb{N}$;
Begin
  $y := 0$;
  $\sigma_0 := \varepsilon$;
  For[1] $j := 0$ **To** $+\infty$ **Do Begin**
    For[2] $(q, \ell_0, \ldots, \ell_m, z) \in \mathbb{L}_{m+3}(y)$ in lexicographical order **Do Begin**
      $\sigma_j := \sigma_0 \diamondsuit (y+1, h_j) \diamondsuit \cdots \diamondsuit (y+q, h_j)$;
      If[1] $j = k$ **And** $y + 1 \leq \langle k, x\rangle \leq y + q$ **Then Return** h_j;
      If[2] $\{\mathcal{M}(\sigma_j)\}(\ell_0)\downarrow^z$ **And** ... **And** $\{\mathcal{M}(\sigma_j)\}(\ell_m)\downarrow^z$ **Then Exit**[2]
    End[2];
    $y' := \max\{y + q, \ell_m\}$;
    $\sigma_j := \sigma_j \diamondsuit (y + q + 1, h_j) \diamondsuit \cdots \diamondsuit (y', h_j)$;
    For[3] $i : y < i \leq y'$ **Do**
      If[3] $i \notin \{\ell_0, \ldots, \ell_m\}$ **Then** $\sigma_0(i) := \sigma_j(i)$
        Else If $\phi_{\mathcal{M}(\sigma_j)}(i) \neq h_0$ **Then** $\sigma_0(i) := h_0$ **Else** $\sigma_0(i) := \sigma_j(i)$;
    For[4] $s := 1$ **To** $j - 1$ **Do** $\sigma_s := \sigma_s \diamondsuit (y + 1, \widehat{\sigma}_0(y + 1)) \diamondsuit \cdots \diamondsuit (y', \widehat{\sigma}_0(y'))$;
    If[4] $\langle k, x\rangle \leq y'$ **And** $k < j$ **Then Return** $\widehat{\sigma}_k(\langle k, x\rangle)$;
    $y := y'$;
  End[1]
End

Figure 7: A recursive Ξ functional. The "For[2]" loop is performed in the lexicographical order. $\mathbb{L}_{m+3}(y) = \{(q, \ell_0, \ldots, \ell_m, z) \in \mathbb{N}^{m+3} : y < q < \ell_0 < \cdots < \ell_m < z\}$.

Let $\mathcal{M}$ be any (total) scientist. With the help of the functional specified in Figure 7 and Proposition 9, we show that the scientist $\mathcal{M}$ cannot Bc^n-identify some functions in $\mathcal{S}^{n+1}$. Let $\mathbb{L}_{m+3}(y) = \{(q, \ell_0, \ldots, \ell_m, z) \in \mathbb{N}^{m+3} : y < q < \ell_0 < \cdots < \ell_m < z\}$, a set of $(m+3)$-ordered tuples of positive integers. The value of q refers to a step extension of the domain of the functions in construction; the values ℓ_0, ..., ℓ_m are tentative points of convergence of some function of code $\mathcal{M}(\sigma_j)$, for $\sigma_j \in SEG$; finally the value z is the number of steps of computation of program code $\mathcal{M}(\sigma_j)$ allowed in the current tentative of convergence on the inputs ℓ_0, ..., ℓ_m. The following are the main characteristics of the specification where it is assumed that scientists are total:

1. The elements σ_0, σ_1, ..., σ_j of SEG are all prefixes of graphs of functions; each prefix σ_k, from $k = 0$ to $k = j$ is extended in each step of the external "For[1]" loop, whenever the "For[2]" loop is interrupted. The programme code $\mathcal{M}(\sigma_j)$ is executed z steps on inputs ℓ_0, ..., ℓ_m; eventually, for some tuple $(q, \ell_0, \ldots, \ell_m, z)$, the search is successful and the "For[2]" loop is interrupted. After the successive executions of the "For[2]" loop, the domain of function $\widehat{\sigma}_0$ is extended in the "For[3]" loop, from $\{0, \ldots, y\}$ to $\{0, \ldots, \max\{y + q, \ell_m\}\}$. The "For[4]" loop makes all the functions $\widehat{\sigma}_k$ so far defined (for $k = 1$ to $k = j - 1$) to follow the values of $\widehat{\sigma}_0$. Note that, whenever the "For[2]" loop is non-terminating at final step j, σ_0, σ_1, ..., σ_{j-1} are graphs of functions with finite domain, but $\lim \sigma_j$ will always be a text for a total function.

2. If the "For[2]" loop does not succeed at some final step j, even in the

presence of a total function h given as input, then ϕ_{h_j} comes to be a total function in $\mathcal{S}$, with constant value h_j from some order on, for which the scientist $\mathcal{M}$ fails to converge:

$$\mathcal{S}^{n+1} \supset \mathcal{S} \ni \phi_{h_j} \quad = \quad h_0 \quad \underbrace{h_1 \cdots h_{j-1}}_{\text{increasing repeated values}} \quad h_j h_j \cdots h_j \cdots$$

In this case, the scientist $\mathcal{M}$ fails to Bc^n-identify the function $\phi_{h_j} \in \mathcal{S} \subseteq \mathcal{S}^{n+1}$.

3. If the "For[2]" loop always succeeds and the function h is total, then, for every $j \in \mathbb{N}$, $\lim \widehat{\sigma}_j$ comes to be a total function. Note that $\lim \sigma_0$ is defined under the clause "For[3]" and that $T = \lim \sigma_0$ is a text for the function $\Xi(h)(\langle k, x\rangle)$. In this case, all the values of the total increasing function h, such that, for every $k \in \mathbb{N}$, $\phi_{h_k}(x) = \Xi(h)(\langle k, x\rangle)$, are $(m+1)$-variants of the code of h itself. For some order x_0, for every $x \geq x_0$, $\mathcal{M}(T[x])$ provides codes h_j of the function $\lim \widehat{\sigma}_0$ such that $\phi_{h_j} = \lim \widehat{\sigma}_j$ differs from h in $m+1$ input values, meaning that the scientist $\mathcal{M}$ is not able to Bc^n-identify the function $\lim \widehat{\sigma}_0 \in \mathcal{S}^{n+1}$, given the text $T = \lim \sigma_0$. In another words, we can say that, in this case, for every i and j, $i \neq j$, $\lim \widehat{\sigma}_i$ and $\lim \widehat{\sigma}_j$ are $(m+1)$-variants.

As a corollary of Proposition 13, we have that

Proposition 14 *(Case and Smith [9, 10], Jain et al. [15])* $\mathcal{S}^\star \in (Bc^\star - \cup_{n\in\mathbb{N}} Bc^n)$.

It is straightforward to conclude that $\mathcal{S}^\star \in Bc^\star$. Let us suppose that $\mathcal{S}^\star \in \cup_{n\in\mathbb{N}} Bc^n$. Then, there exists a k such that $\mathcal{S}^\star \in Bc^k$. In particular we have $\mathcal{S}^k \in Bc^k$ and $\mathcal{S}^{k+1} \in Bc^k$ which contradicts Proposition 13.

12 Does it end at $Bc^\star$?

Taking into consideration the last result of the previous section, we conclude the following consequence of the fact that $Bc^\star - Bc^n \neq \emptyset$, for any $n \in \mathbb{N}$:

Proposition 15 $\mathcal{R} \notin Bc^n$, *for any* $n \in \mathbb{N}$.

It follows again that, accepting anomalies, this time varying from hypothesis to hypothesis, increases the power of scientific inference, in an infinite non-collapsing hierarchy of classes of sets:

Proposition 16 $Bc^0 \subset Bc^1 \subset Bc^2 \subset \cdots \subset Bc^n \subset \cdots \subset Bc^\star$.

Scientist $\mathcal{M}(\sigma[t] : SEG; x : \mathbb{N}) : \mathbb{N}$;
Begin
  For $m := 0$ **To** $|\sigma|$ **Do Begin**
    $\tau = \emptyset$;
    Decode m in its programme form $\{m\}$;
    % If programme m, on input i, does not halt after x transitions,
      % then $\{m\}(i)\downarrow^{\leq x} = \perp$;
    For $i := 0$ **To** $|\sigma| - 1$ **Do** $\tau := \tau \diamondsuit (i, \{m\}(i)\downarrow^{\leq x})$;
    If $\tau = \sigma$ **Then Return** $\{m\}(x)$
  End;
  Return 0
End

Figure 8: This scientist searches for the code of a function provided by the succession of prefixes of its graph.

Given data $\psi[t] = (0, \psi(0))(1, \psi(1)) \ldots (t-1, \psi(t-1))$ and a single point x, we now specify how to evaluate $\psi(x)$, for arbitrarily large x.

Let $\{m\}(i)\downarrow^{\leq x}$ denote the output of the programme code m on input i, if it halts within x steps of computation; otherwise $\{m\}(i)\downarrow^{\leq x}$ gives some indetermination symbol.

A scientist $\mathcal{M}$ can be assembled as in Figure 8. Applying to this function the $s-1-1$-theorem we find an index $s(\sigma[t])$, where s is a (total) recursive function, such that $\phi_{s(\sigma[t])}(x) = f(\sigma[t], x)$.[11] We claim that there is an order p, such that, for $t \geq p$, $s(\sigma[t])$ is a $\star$-variant code for the recursive function ψ:

1. The code of the wanted ψ with prefix σ is certainly between 0 and a sufficient large value of $t = |\sigma|$ (the number of function pairs in σ); however, even for sufficiently large t, convergence after x steps is only guaranteed for sufficient large x.

2. For sufficient large values of x, the scientist has the time to witness $\{m\}(i)\downarrow^{\leq x}$, for some m $(1 \leq m \leq t)$, converging in all entries i $(0 \leq i \leq t-1)$;

3. Then for sufficient large values of t, the function $\phi_{s(\sigma[t])}$ coincides with the wanted ψ, up to a finite collection of errors, i.e. $\phi_{s(\sigma[t])} = \psi$ for all x greater than some order p;

4. Then the code $s(\sigma[t])$ is a $\star$-variant code of ψ and it is computable;

5. As t and x increases, depending on σ, even in the cases where scientist conjectures a correct code, the code e of the $\star$-variant of the function ψ may vary; the reason for this continuous mind change is due to the fact that, for each t, the value of x at which m converges for all entries in $\sigma[t]$ varies with t;

6. Any recursive function becomes $Bc^{\star}$-identifiable;

[11] Note that the sequences $\sigma \in SEG$ can be encoded into the natural numbers as well.

7. The $Bc^\star$-identification is done by the computable scientist which on input $\sigma[t]$ outputs de code $s(\sigma[t])$.

In this way, we find scientists witnessing that $\mathcal{R} \in Bc^\star$, that is:

Proposition 17 *(Leo Harrington cited in Case and Smith [10], Jain et al. [15])*
$\mathcal{R} \in Bc^\star$.

Reaching $\mathcal{R}$, the universe of functions from where we started, we reach *totality*, i.e. the full power of scientists. The class $Bc^\star$ is then the maximum expressive power in identification tasks.

Proposition 17 says that computable scientists have the power to distinguish all (total) computable relations between physical magnitudes, if they permit a finite, yet unlimited, but variable from hypothesis to hypothesis, number of errors.

Once again, we recall that algorithmic scientific inference is based on asymptotic behaviour of functions, and that unlimited but finite number of errors means a zero density of errors on the overall behaviour of each $\star$-variant of the function. We conclude the ultimate paradigm that do for $\mathcal{R}$.

CRITERION 4: UNCONVENTIONAL SCIENTISTS : A scientist converge semantically to a law with exceptions, that is from some order k on, the scientist conjectures numbers $p_k, p_{k+1}, \ldots$, such that, for all $i \geq k$, ϕ_{p_i} is a correct version of law up to finitely many omission errors or exceptions, not necessarily in the same points.

13 Conclusion

We postulated a computable universe where the laws that rule observations and measurements can be written as recursive relations. We review the way how these laws can be algorithmically established in the limit. A very simple algorithm can be used to learn all recursive functions if the unconventional scientist permits a finite number of errors or exceptions and accept to converge semantically to the laws. If the scientist does not accept laws with exceptions, then the universe she can knows with a single universal method is a strict subset of the recursive relations. Finally, if the universal method is supposed to converge syntactically to final conjectures, then the the conventional scientist becomes more limited in what she can learn.

The reader may question why to postulate that empirical laws are recursive or even just a subset of the recursive functions. There is a formal reason. Let us assume that some scientist observes the pulses of a bulb of light switching between modes "on" and "off", say 1 or 0, respectively. Let the time between consecutive *bits* be of 1 second. A priori, we may think that any function from time instants in $\mathbb{N}$ to the set $\{0, 1\}$ is permitted, so that the universe of possible sequences is $\mathcal{K} = 2^{\mathbb{N}}$. The scientist start observing the sequence of

bits and conjectures a possible law, at any new *datum*, at any new observation. Historically, the syntax of rules used in the specification of empirical laws is rather elementary. More generally, it would have a fully recursive grammar. In fact there is no other way of doing it, since a non-recursive relation cannot be expressed by finite means. Let us assume that the most powerful scientist [12] conjectures the least code of a recursive relation that mimics the sequence of bits so far observed. If the sequence of bits is non-recursive, then the scientist will be changing her mind infinitely often. However, if the scientist converges to a code number then the sequence is recursive. Thus the scientist cannot decide in the limit if the observations are recursive. However, the scientist can verify in the limit (but not refute in the limit) the hypothesis that the observations are recursive. Reciprocally, we can also conclude that the scientist can refute in the limit if the observations are non-recursive, although she cannot refute in the limit that the observations are recursive. Thus cognition of a scientific law is verifiable in the limit but not refutable in the limit. Kevin Kelly in [16] dedicates many pages of his book to convince the reader that the model universe should be computable. So far it seems that scientific empiric laws are expected to be of a recursive nature.

We also provided evidence from Machine Learning Theory that attempts to go further than the primitive recursive relations [13] with a single universal method imply a paradigm change on the character of a scientific law, in the sense that a law with exceptions is not traditionally considered at the same level of other scientific laws, but they have to be promoted as such in the view of an enlargement of the class of empirical laws.

We also revived some criticism against Popper's refutability in order to promote a kind of non-rational method in learning a scientific law.

Our proofs are based in the original formulation of Case and Smith [9, 10], presented also in Osherson et al. book [15] and by Odifreddi in [19]. Some constructions in previous work were found to be implicit applications of Case's Theorem in [7]. We adopted a explicit view of building up the functionals used in the proofs, remaking the framework in this light.

Acknowledgements. The research of José Félix Costa is supported by Fundação para a Ciência e Tecnologia, projeto FCT I.P.:UID/FIL/00678/2013. The author is thankful to John Case for the motivation to invest in Learning Theory and to Bruno Patrício for his reading and comments.

References

[1] Janis Bārzdiņš. Inductive inference of automata, functions and programs. In *International Mathematical Congress, Vancouver*, volume II, pages 455–560. Latvian State University, 1974.

[12] Since although the scientist cannot see the bits to come, she can solve the halting problem.

[13] Those that can be expressed computationally by programs that are build on top of arbitrary number of nested for loops

[2] Lenore Blum and Manuel Blum. Toward a mathematical theory of inductive inference. *Information and Control*, 28:125–155, 1975.

[3] Lenore Blum, Felipe Cucker, Michael Shub, and Steve Smale. *Complexity and Real Computation*. Springer, 1998.

[4] Olivier Bournez and Emmanuel Hainry. Elementarily computable functions over the real numbers and $\mathbb{R}$-sub-recursive functions. *Theoretical Computer Science*, 348(2–3):130–147, 2005.

[5] Olivier Bournez and Emmanuel Hainry. Recursive analysis characterized as a class of real recursive functions. *Fundamenta Informaticae*, 74(4):409–433, 2006.

[6] Manuel Campagnolo, Cris Moore, and José Félix Costa. An analog characterization of the Grzegorczyk hierarchy. *Journal of Complexity*, 18(4):977–1000, 2002.

[7] John Case. Infinitary self-reference in learning theory. *Journal of Experimental and Theoretical Artificial Intelligence*, 6(1):3–16, 1994.

[8] John Case. Algorithmic scientific inference: Within our computable expected reality. *International Journal of Unconventional Computing*, 8(3):192–206, 2012. Invited journal expansion of an invited talk and paper at the *3rd International Workshop on Physics and Computation 2010*.

[9] John Case and Carl Smith. Anomaly hierarchies of mechanized inductive inference. In Richard J. Lipton, Walter A. Burkhard, Walter J. Savitch, Emily P. Friedman, and Alfred V. Aho, editors, *Proceedings of the 10th Annual ACM Symposium on Theory of Computing, May 1-3, 1978*, pages 314–319. ACM, San Diego, California, USA, 1978.

[10] John Case and Carl Smith. Comparison of identification criteria for machine inductive inference. *Theoretical Computer Science*, 25(2):193–220, 1983.

[11] R. Daley. On the error correcting power of pluralism in BC-type inductive inference. *Theoretical Computer Science*, 24(?):95–104, 1983.

[12] E. M. Gold. Language identification in the limit. *Information and Control*, 10:447–474, 1967.

[13] Jr. Hartley Rogers. *Complexity and Real Computation*. MIT Press, third edition, 1987. First edition published in 1967 by McGraw Hill.

[14] Robert Hecht-Nielsen. *Neurocomputing*. Addison-Wesley Publishing Company, 1990.

[15] 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.

[16] Kevin T. Kelly. *The Logic of Reliable Inquiry*. Oxford University Press, 1996.

[17] 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.

[18] Albert Meyer and Dennis Ritchie. The complexity of loop programs. In *Proceedings of the 22nd National Conference*, pages 465–470. Thompson Book Company, 1967.

[19] Piergiorgio Odifreddi. *Classical Recursion Theory II*. Studies in Logic and the Foundations of Mathematics. North Holland, 1999.

[20] Karl R. Popper. *The Logic of Scientific Discovery*. Routledge, 1935. First English edition published in 1959 by Hutchinson & Co. First published by Routledge in 1992.

[21] 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.