# An analytic condition for $P \subset NP$

José Félix Costa * Jerzy Mycka †

March 23, 2006

**Abstract**

In this paper, we prove that there exists some condition, involving real functions, which implies $P \neq NP$.

## 1 Introduction and motivation

The theory of analog computation, where the internal states of a computer are continuous rather than discrete, has enjoyed a recent resurgence of interest. In this historical framework we go back to Claude Shannon's (see [17]) so-called General Purpose Analog Computer (GPAC). This was defined as a mathematical model of an analog device, the Differential Analyser, the fundamental principles of which were described by Lord Kelvin in 1876. We have been working recently towards inductive definitions of generalized GPAC functions over $\mathbb{R}$.

The first presentation of such a theory, analogous to Kleene's classical theory of recursive functions over $\mathbb{N}$, was attempted by Cristopher Moore [10]. Real recursive functions are generated by a fundamental operator, called differential recursion. (The other fundamental operator is the taking of infinite limits, introduced in [11].) Let us recall the concept of *a real recursive function*, and the corresponding class $REC(\mathbb{R})$, as was introduced in [12]. The class $REC(\mathbb{R})$ of real recursive vectors is generated from the real recursive scalars 0, 1, and $-1$, and the real recursive projections, by the following operators: composition, differential recursion (the solution of a Cauchy problem or a initial value problem in mathematical analysis), and infinite limits.

If we consider a wider context for solving differential equations, then it is possible to justify the concept of *generalized solution* of a given differential equation. We present in this paper the concept of *(restricted) real recursive function* and the corresponding class $REC_R(\mathbb{R})$, which is based on the above definition of real recursive function, without infinite limits, with a new understanding of differential recursion.

*Department of Mathematics, I.S.T., Universidade Técnica de Lisboa, Lisboa, Portugal, fgc@math.ist.utl.pt

†Institute of Mathematics, University of Maria Curie-Sklodowska, Lublin, Poland, Jerzy.Mycka@umcs.lublin.pl, corresponding author

With these notions and the additional operator of bounded quantification (to define the concept of *non-deterministic computation*) we are able to introduce some important subclasses of real recursive functions. They are obtained by an imposition of restrictions on the growth of functions from $REC_R(\mathbb{R})$. Because the growth of values of functions is somehow restricted by time of computation such kind of restriction is connected with the complexity of real functions. But let us stress the fact that it is not our purpose to build such classes, which strictly inherit the properties of the classical complexity classes — rather, it is more important for us to have analog classes with robust analytical definitions.

A real recursive function is said to be of exponential order if in any step of its construction, its components are bounded by the exponential function. It is said to be of subexponential order if in any step of its construction, its components are subexponentially bounded. Let us mention that these properties can be clearly presented in mathematical analysis: subexponentially bounded functions can be introduced by the condition that for every such function $f$, its Laplace transform $L[f](s)$ is defined along the whole positive real axis $\Re\ s > 0$; in the exponential case we are interested only in the existence of the Laplace transform.

By using of weak exponential restrictions ($e^{(log x)^k}$) we obtain the classes $DAnalog$ and $NAnalog$. We can also use a notion of admissible restrictions to transform real functions into natural ones not exceeding barriers of polynomial complexity.

These classes have many nice properties. For example, $DAnalog$ and $NAnalog$ can be simply related to linear differential equations. Moreover, many well known physical systems are in $DAnalog$ (like $RC$ circuits, $RLC$ circuits, harmonic oscillators). The use of Laplace transforms makes $DAnalog$ really meaningful and natural, since physical measures in physical theories do have the Laplace transform, having then a controlled growth. In this sense our complexity classes have physical meaning.

Then we can prove that the condition $P = NP$ implies the identity of the above mentioned restrictions of $DAnalog$ and $NAnalog$. By contraposition we obtain an analytical test for the conjecture $P \neq NP$.

This paper is an attempt to use the above approach for the conjecture $P \neq NP$. With respect to [13], Section 2 introduces a more clear formulation of a (discrete) condition equivalent to $P = NP$, clarifying aspects that were obscure in [13], Section 3 is completely new and considers a generalized solution to the differential recursion scheme, Section 4 indeed intersects [13], but contains different definitions and modified concepts, capitalizing in some results from the precedent paper, and the new Section 5 motivates strongly our analog classes, relates them with linear systems, and with solutions of a feasible physics. In this Section, the conjecture $P \subset NP$ is lifted to the realm of Analysis.

With quantifier elimination (using techniques from the sixties, namely the concept of Richardson's map in [16]) in the non-deterministic class $NAnalog$, hopefully, we will succeed in the near future to present a quantifier free (full) analytic condition for the conjecture $P \subset NP$.

## 2 Polynomial time computable functions

The main model of computation, the Turing machine, has an obvious correspondence with the class of recursive functions. We can distinguish within the set of recursive functions a subset $P\mathcal{F}$ corresponding to deterministic Turing machines working in polynomial time.

**Definition 1** *A partial function $f$ is said to be* computable *by a deterministic Turing machine $M$ if (a) $M$ accepts the domain of $f$ and (b) if $\langle x_1, \ldots, x_n \rangle \in$* dom(f)*, then the accepting computation writes in the output tape the value $f(x_1, \ldots, x_n)$.*

**Definition 2** *$P\mathcal{F}$ is the class of partial functions that can be computed in polynomial time by deterministic Turing machines, i.e., by deterministic Turing machines clocked with polynomials.*

We adopted the definition of computable function given in [2]. Note that for functions in $P\mathcal{F}$ the *halting problem* is decidable. We have an inductive definition of the total functions in $P\mathcal{F}$, provided by Buss in 1986 (see, e.g., [15], p. 172):

**Proposition 3** *The class of* recursive functions computable in deterministic polynomial time *is inductively defined from the basic functions $Z = \lambda n.\ 0$ and $S = \lambda n.\ n+1$, the projections, basic functions $\lambda n.\ 2n$, $\lambda n.\ 2n+1$, $\lambda n.\ \lfloor \frac{n}{2} \rfloor$, the characteristic function $\delta$ of "equality to 0", by the operations of composition, definition by cases, and polynomially bounded primitive recursion.* □

The intuition behind this characterization of the total functions in $P\mathcal{F}$ is the following: a Turing machine clocked in polynomial time $p$ can write at most $p(|x|)$ bits in the output tape for the input $x$ ($|x|$ is the length of $x$). This number is bounded by $2^{\lfloor \log(x+1) \rfloor^k}$, for some $k$.

We define now, for our purpose, a non-deterministic Turing machine in a way similar to which it is used in the probabilistic computational model (compare with [2]). We impose the following conditions on the non-deterministic machines: (a) every step of a computation can be made in exactly two possible ways, which are considered different even if there is no difference in the corresponding actions (this distinction corresponds to two different bit guesses), (b) the machine is clocked by some time constructible function and the number of steps in each computation is exactly the number of steps allowed by the clock; if a final state is reached before this number of steps, then the computation is continued, doing nothing up to this number of steps, (c) every computation ends in a final state, which can be either *accept* or *reject*, (d) if the machine computes a function, then all accepting computations write down to the output tape the value of the function (see [2], Chapters 3 and 6 for a comparison). It is irrelevant what the machine writes in the output tape if it reaches a rejecting state.

**Definition 4** *A partial function $f$ is said to be* computable *by a non-deterministic Turing machine $M$ if (a) $M$ accepts the domain of $f$ and (b) if $\langle x_1, \ldots, x_n\rangle \in \mathrm{dom(f)}$, then any accepting computation writes in the output tape the value $f(x_1, \ldots, x_n)$.*

**Definition 5** *$NP\mathcal{F}$ is the class of partial functions that can be computed in polynomial time by non-deterministic Turing machines, i.e., by non-deterministic Turing machines clocked with polynomials.*

**Proposition 6** *$NP\mathcal{F}$ is the class of functions of the following form:* $\lambda\langle x_1, \ldots, x_n\rangle$. if $\langle x_1, \ldots, x_n\rangle \in A$ then $F(x_1, \ldots, x_n)$, *where $A \in NP$ and $F \in P\mathcal{F}$.*

**Proof.** Let $f$ be a function of the above given form. Let $M$ be a non-deterministic Turing machine that recognizes $A$ in $NP$ and $N$ a deterministic Turing machine that computes $F$ in $P\mathcal{F}$. The following non-deterministic Turing machine computes $f$:

**procedure:**
**begin**
input $\langle x_1, \ldots, x_n\rangle$;
simulate non-deterministically $M$ on $\langle x_1, \ldots, x_n\rangle$;
simulate $N$ on $\langle x_1, \ldots, x_n\rangle$;
if $M$ and $N$ accept then output the result of $N$
**end**

Conversely, let $f$ be a function in $NP\mathcal{F}$ and $M$ a non-deterministic Turing machine that computes $f$ in polynomial time $p$. Let Guess(M) $= \{\langle x_1, \ldots, x_n, z\rangle : z$ guides $M$ with input $\langle x_1, \ldots, x_n\rangle$ to an accepting state$\}$. The following procedure first makes a test in $NP$ and then computes a function in $P\mathcal{F}$, with the format $\lambda x$. if $\langle x_1, \ldots, x_n\rangle \in A$ then $F(x_1, \ldots, x_n)$:

**procedure:**
**begin**
input $\langle x_1, \ldots, x_n\rangle$;
guess $|z| \leq p(|\langle x_1, \ldots, x_n\rangle|)$;
if $\langle x_1, \ldots, x_n, z\rangle \in$ Guess(M) then
simulate $M$ on $\langle x_1, \ldots, x_n\rangle$ with guess $z$
and output the result
**end**

□

**Definition 7** *We define the class $\exists P\mathcal{F}$ as follows: a function $f : \mathbb{N}^n \to \mathbb{N}$ is in $\exists P\mathcal{F}$ if there exists a function $F : \mathbb{N}^{n+1} \to \mathbb{N}$ in $P\mathcal{F}$ and a polynomial $p : \mathbb{N}^n \to \mathbb{N}$ such that (a) $\langle x_1, \ldots, x_n\rangle \in \mathrm{dom(f)}$ if and only if there exists a number $k$ such that $|k| \leq p(\sum_{i=1}^{n} |x_i|)$ and $\langle x_1, \ldots, x_n, k\rangle \in \mathrm{dom(F)}$ and (b) $f(x_1, \ldots, x_n)$*

*is defined and* $f(x_1, \ldots, x_n) = y$ *if and only if for the number* $k$ *such that* $|k| \leq p(\sum_{i=1}^{n} |x_i|)$ *and* $F(x_1, \ldots, x_n, k)$ *is defined we have* $F(x_1, \ldots, x_n, k) = y$*, and, moreover for all such* $|k| \leq p(\sum_{i=1}^{n} |x_i|)$ *that* $\langle x_1, \ldots, x_n, k\rangle \in \mathrm{dom(F)}$*, we have* $F(x_1, \ldots, x_n, k) = y$*.*

With the last definition we can present the following two obvious propositions.

**Proposition 8** *The class* $NP\mathcal{F}$ *coincides with* $\exists P\mathcal{F}$*.*

**Proposition 9** $NP \subset P$ *if and only if* $NP\mathcal{F} \subset P\mathcal{F}$*.*

For the purpose of this paper, we adopt the following convention: if $f$ is undefined at $x$, then we take $f(x) = 0$ (adding 1 to a function allows coding for zeros without ambiguity). Since the domain of a function in $P\mathcal{F}$ is decidable in polynomial time, functions computable in deterministic polynomial time can always be considered to be total: let $PF$ be such a class $P\mathcal{F}$ redefined.

**Definition 10** *Let* $PF = \{\tilde{f} : f \in P\mathcal{F}\}$*, where* $\tilde{f}$*, for a* $n$*-ary function* $f$*, is given by cases*

$$\tilde{f}(x_1, \ldots, x_n) = \begin{cases} f(x_1, \ldots, x_n) + 1 & \text{if } \langle x_1, \ldots, x_n\rangle \in dom(f) \\ 0 & \text{otherwise} \end{cases}$$

Although the domain of a function in $NP\mathcal{F}$ may not be decidable in deterministic polynomial time, functions computable in non-deterministic polynomial time can also be considered to be total: let $NPF$ be the class $NP\mathcal{F}$ redefined in this way, whose functions take the value 0 at $x$ whenever they come from functions in $NP\mathcal{F}$ undefined at $x$.

**Definition 11** *Let* $NPF = \{\tilde{f} : f \in NP\mathcal{F}\}$*, where* $\tilde{f}$*, for a* $n$*-ary function* $f$*, is given by cases*

$$\tilde{f}(x_1, \ldots, x_n) = \begin{cases} f(x_1, \ldots, x_n) + 1 & \text{if } \langle x_1, \ldots, x_n\rangle \in dom(f) \\ 0 & \text{otherwise} \end{cases}$$

A function from $NP\mathcal{F}$ can be presented as $\lambda\langle x_1, \ldots, x_n\rangle$*. if* $\langle x_1, \ldots, x_n\rangle \in A$ *then* $F(x_1, \ldots, x_n)$, for some set $A \in NP$ and some $F \in P\mathcal{F}$ and a function from $NPF$ can be proved to be of the form $\lambda\langle x_1, \ldots, x_n\rangle$*. if* $\langle x_1, \ldots, x_n\rangle \in A$ *then* $F(x_1, \ldots, x_n)$ *else* 0, for some set $A \in NP$ and some $F \in PF$. Then we have the main result:

**Proposition 12** $NP \subset P$ *if and only if* $NPF \subset PF$*.*

**Proof.** Assume that $NP \subset P$ and let $f \in NPF$. Then $f$ is of the form $\lambda\langle x_1, \ldots, x_n\rangle$. if $\langle x_1, \ldots, x_n\rangle \in A$ then $F(x_1, \ldots, x_n)$ else 0, with $A \in NP$ and $F \in PF$. We conclude that $A \in P$ and, consequently, $f \in PF$. Conversely, assume that $NPF \subset PF$ and let $A \in NP$. Then the function $f$ defined by $\lambda x$*. if* $x \in A$ *then* 1 *else* 0 is in $NPF$. We conclude that $f \in PF$ and, consequently, $A \in P$. □

The following proposition is rather trivial.

**Proposition 13** *Functions in $NPF$ can not grow faster than functions in $PF$.*

This last proposition means that each function in $NPF$ is bounded by a weak exponential bound (sometimes called a quasi-polynomial) $\lambda x.2^{\lfloor \log(x+1) \rfloor^k}$, for some $k$.

We can redefine $PF$ and $NPF$ in such a way that Proposition 12 still holds and both classes are closed for composition: consider the restrictions of these two classes to strict functions (like in domain theory, with re-interpretation of the undefined value), i.e., to functions that give value 0 whenever one of their arguments is 0. The closure for composition arises from the fact that classes $P$ and $NP$ are closed for the union and intersection of sets. However, although this closure can be elegantly formulated, it does not help in the continuation of this paper.

We also note that we could have substituted $NPF$ by $PF(NP)$, i.e., functions computable in polynomial time with oracles in $NP$. We also have, $P = NP$ if and only if $PF = PF(NP)$. In the next sections we will work with the classes of total functions $PF$ and $NPF$. We will write $NPF = \exists PF$ for $NP\mathcal{F} = \exists P\mathcal{F}$ according with the conventions introduced previously.

# 3 Recursive functions over $\mathbb{R}$

Let us start with examples and considerations, which help us to introduce the notion of a real recursive function. We will use the concept of *a vector function* to denote a real function from $\mathbb{R}^k$ to $\mathbb{R}^n$, of *a scalar function* to denote a real function from $\mathbb{R}^k$ to $\mathbb{R}$.

*Differential recursion* is an important operator for the theory of analog computation (and especially for real recursive functions). It is based on the following scheme of differential equations. If $f$ is a vector function with $n$ $k$-ary components and $g$ is a vector function with $n$ $(k+n+1)$-ary components, then we can define the new vector function $h$ of $n$ $(k+1)$-ary components which is the solution of the Cauchy problem, $1 \leq i \leq n$,

$$h_i(x_1, \ldots, x_k, 0) = f_i(x_1, \ldots, x_k),$$

$$\partial_y h_i(x_1, \ldots, x_k, y) = g_i(x_1, \ldots, x_k, y, h_1(x_1, \ldots, x_k, y), \ldots, h_n(x_1, \ldots, x_k, y)).$$

But there is a question of restrictions and conditions which should be placed on the solution to obtain a robust and clear theory. Examples that follow along this Section are adapted from [3].

**Example 14** *Consider the scheme $h(0) = 1$, $\partial_y h(y) = \frac{h(y)^2 - 1}{y^2 + 2y}$. Both $h(y) = 1$ and $h(y) = y + 1$ are solutions to this equation. The problem is that $\partial_y h(y)$ is not defined at $y = 0$. Since the solution is not unique we cannot define a function by differential recursion out of $f = 1$, $g(y, z) = \frac{z^2 - 1}{y^2 + 2y}$.*

Note that, in differential recursion, it is not imposed that functions $f_i$ and $g_i$, for $1 \leq i \leq n$, are of class $C^1$. To emphasize this important fact we give an example.

**Example 15** *A fruitful example is the differential scheme* $h(0) = 2 - \sqrt{3}$, $\partial_y h(y) = \frac{y}{h(y)-2}$, *where the functions* $f$ *and* $g$ *are the constant* $2 - \sqrt{3}$ *and* $\lambda yz.\ \frac{y}{z-2}$, *respectively. Although the function* $g$ *is not* $C^1$ *in all components, the solution is* $h(y) = 2 - \sqrt{y^2 + 3}$, *defined in all* $\mathbb{R}$.

What is a solution of our differential recursion scheme? Books on differential equations (e.g. [9]) say that *a solution... is a function of the independent variable that, when substituted into the equation as the dependent variable, satisfies the equation for all values of the independent variable.* That is, a function $h(y)$ is a solution if it satisfies $\partial_y h(y) = g(y, h(y))$, for all $y$ in $\mathbb{R}$. But in many cases, we have that there is a unique function $h$ in $C^1$ that satisfies the equation for all $y$ where $g$ is defined, although $g$ has a countable number of discontinuities in, e.g., $\mathbb{R}$. In this case we can adopt $h$ as the desired solution of the differential recursion scheme. We want to consider types of differential recursion schemata having these kinds of solutions. It will make our theory more simple. For that purpose we will allow these solutions for a countable number of possible discontinuities of $g$.

**Example 16** *Consider the scheme* $h(0) = 0$, $\partial_y h(y) = \frac{1}{\sec(y)}$. *Classically the solution is* $h(y) = \sin(y)$, *e.g, for* $y \in (-\frac{\pi}{2}, \frac{\pi}{2})$. *But, if we ask for the maximal generalized solution in* $C^1$ *or even in* $C^0$, *the answer is* $h(y) = \sin(y)$ *in* $\mathbb{R}$, *despite the fact that our differential relation will only be satisfied outside a countable number of points.*

**Example 17** *Consider the scheme* $h(0) = 0$, $\partial_y h(y) = \frac{h(y)}{y}$. *Classically the solution is* $h(y) = y$, *but either for* $y \in (-\infty, 0)$ *or for* $y \in (0, +\infty)$. *But, if we ask for the maximal generalized solution in* $C^1$ *or even in* $C^0$, *the answer is* $h(y) = y$, *which means that this solution is the unique continuous continuation in* $\mathbb{R}$.

Now as the results of the above remarks we can propose the following definition.

**Definition 18** *A solution to a system of equations (for* $1 \leq i \leq n$*) given by the following differential recursion*

$$\partial_y h_i(x_1, \ldots, x_k, y) = g_i(x_1, \ldots, x_k, y, h_1(x_1, \ldots, x_k, y), \ldots, h_n(x_1, \ldots, x_k, y)),$$

*satisfying the initial conditions* $h_i(x_1, \ldots, x_k, 0) = f_i(x_1, \ldots, x_k)$ *is a vector function* $\hat{h}$ *from* $\mathbb{R}^{k+1}$ *to* $\mathbb{R}^n$ *such that:*

- *A unique solution* $h$ *to the system of differential equations exists in some open interval* $I$ *containing* $0$*;*
- *The vector function* $\hat{h}$ *satisfies the equations in a set* $J \supseteq I$, *such that* $J$ *is an open interval up to a countable number of non-Zeno discontinuities,*[1] *in*

[1] It means that for each finite open interval there exist only a finite number of discontinuities.

*the sense that, for every* $y \in J$, $\hat{h}_i(x_1, \ldots, x_k, y)$, $g_i(x_1, \ldots, x_k, y, \hat{h}_1(x_1, \ldots, x_k, y), \ldots, \hat{h}_n(x_1, \ldots, x_k, y))$, *and* $\partial_y \hat{h}_i(x_1, \ldots, x_k, y)$ *are defined, for all* $1 \leq i \leq n$, *and it holds that*

$$\partial_y \hat{h}_i(x_1, \ldots, x_k, y) = g_i(x_1, \ldots, x_k, y, \hat{h}_1(x_1, \ldots, x_k, y), \ldots, \hat{h}_n(x_1, \ldots, x_k, y));$$

- *The vector functions* $h$ *and* $\hat{h}$ *coincide on* $I$*;*
- *The vector function* $\hat{h}$ *is the unique* $C^1$ *extension of* $h$*;*
- $J$ *is the largest such set.*

The vector function $\hat{h}$ is called *the generalized solution* of the system if it is the maximal solution according with the previous items. When dealing with our inductive definitions, we will work with this definition of a solution to the differential recursion scheme. Some informal aspects of such a definition were implicitly developed with one of the authors in [5] but never explicitly formalized.

With this operator we can present the concept of *(restricted) real recursive function* and the corresponding class $REC_R(\mathbb{R})$ (based on the similar definition from [12]).

**Definition 19** *The class* $REC_R(\mathbb{R})$ *of real recursive vector functions is generated from the real recursive scalars* $0$, $1$, $-1$, *the real recursive projections* $I^i_n(x_1, \ldots, x_n) = x_i$, $1 \leq i \leq n$, $n > 0$, *and the real recursive functions* $\theta_k(x) = x^k \Theta(x)$, *for* $k \geq 0$, *by the following operators:*

Composition: *if* $f$ *is a real recursive vector function with* $n$ $k$*-ary components and* $g$ *is a real recursive vector function with* $k$ $m$*-ary components, then the vector function with* $n$ $m$*-ary components,* $1 \leq i \leq n$,

$$\lambda x_1 \ldots \lambda x_m.\ f_i(g_1(x_1, \ldots, x_m), \ldots, g_k(x_1, \ldots, x_m))$$

*is real recursive.*

Differential recursion: *if* $f$ *is a real recursive vector function with* $n$ $k$*-ary components and* $g$ *is a real recursive vector function with* $n$ $(k + n + 1)$*-ary components, then the vector function* $h$ *of* $n$ $(k + 1)$*-ary components which is the solution of the Cauchy problem,* $1 \leq i \leq n$,

$$h_i(x_1, \ldots, x_k, 0) = f_i(x_1, \ldots, x_k),$$

$$\partial_y h_i(x_1, \ldots, x_k, y) = g_i(x_1, \ldots, x_k, y, h_1(x_1, \ldots, x_k, y), \ldots, h_n(x_1, \ldots, x_k, y))$$

*is real recursive whenever* $h$ *is a solution to the differential equation in the sense of Definition 18.*

Assembling and designating components: *(a) arbitrary real recursive vector functions can be defined by assembling scalar real recursive function components into a vector function; (b) if* $f$ *is a real recursive vector function, than each of its components is a real recursive scalar function.*

Let us give some examples of functions generated with the above definition.

**Proposition 20** *The scalar functions* $+$, $\times$, $-$, exp, sin, cos, $\lambda x.\ \frac{1}{x}$, $/$, log, $\lambda xy.\ x^y$ *are real recursive functions.*

**Proof.** See, e.g., [12], where many other examples can be found. □

Particularly interesting real recursive functions given by the iteration operator. There are, since the work of Branicky (see [4]), many ways to simulate in continuous-time the iteration of a discrete-time function. Here we state the result for real recursive functions due to [10, 6, 12].

**Proposition 21** *If* $f$ *is a real recursive scalar total function of arity* $m$*, then the iteration of* $f$ *given by* $F$ *of arity* $(m+1)$*, such that, for all* $n \in \mathbb{N}$*,* $F(n, x_1, \ldots, x_m) = f^n(x_1, \ldots, x_m)$ *is a real recursive scalar too.*

**Definition 22** *If* $f : \mathbb{N}^k \to \mathbb{N}$ *is a primitive recursive function, then a real-valued function* $F : \mathbb{R}^k \to \mathbb{R}$ *is called a* canonical extension *of* $f$ *if*

1. $F$ *is a real recursive,*
2. *for all* $x_1, \ldots, x_k \in \mathbb{N}$*, we have* $F(x_1, \ldots, x_k) = f(x_1, \ldots, x_k)$*,*
3. *for all* $x_i \in [n_i, n_i + 1]$*,* $n_i \in \mathbb{N}$*,* $1 \le i \le k$*, we have* $\min(F(n_1, \ldots, n_k),\ F(n_1 + 1, \ldots, n_k + 1)) \le F(x_1, \ldots, x_k) \le \max(F(n_1, \ldots, n_k),\ F(n_1 + 1, \ldots, n_k + 1))$*.*

**Proposition 23** *If* $f : \mathbb{N}^k \to \mathbb{N}$ *is a primitive recursive function, then* $f$ *has a canonical real recursive extension* $F : \mathbb{R}^k \to \mathbb{R}$*.*

**Proof.** Any primitive recursive function can be defined from the functions $Z$, $S$, $I_n^i$ by composition and recurrence. The proof of our proposition is given by structural induction. The analogues of the basic functions are trivially constructible (see, e.g., [8]). Also constructible are the analogues of functions to code and decode pairs of non-negative numbers.

We use here functions of minimal arity, which do not influence the essence of the proof. Let we consider composition. If $f(x) = h(g(x))$, then the functions $g$ and $h$ have canonical extensions $G$ and $H$ (by induction): $H \circ G$ is chosen to be the canonical extension of $f$.

Now we consider the case when $f$ is defined by recurrence from the pair of functions $g$ and $h$:

$$f(x, 0) = g(x),\ f(x, y + 1) = h(x, y, f(x, y)).$$

Let us remind that $g$ and $h$ have canonical extensions $G$ and $H$ (by induction). We use iteration instead of recurrence (see, e.g., [14]). Let $t : \mathbb{N}^3 \to \mathbb{N}^3$ be such that $t(n, m, k) = (n, m + 1, h(n, m, k))$. Then $t^m(n, 0, g(n)) = (n, m, h(n, m))$.

Hence, from the expression $w(n, k_1, k_2, k_3)$ defined as $t^n(k_1, k_2, k_3)$, we obtain $f(n, m) = I_3^3(w(m, n,\ 0, g(n)))$.

Now it is sufficient to prove that $w$ has a canonical extension, where $t$, as a vector defined from $I_3^1$, $S \circ I_3^2$, $h$, has a canonical extension $T$. We can use the construction of the iteration from Proposition 21.

Let us define $W(s, x, y, z) = u(2s)$, where $s \in \mathbb{R}$ and $W$ is the canonical extension of $w$. Let us observe that derivatives $\partial_t u(t)$ and $\partial_t v(t)$, introduced in proposition 21 (see [12]), have constant sign in the intervals $[n, n+1]$. Hence $u$ and $v$ are monotonic or constant in these intervals. This fact gives $u(2n-2) \leq u(x) \leq u(2n)$, for all $x \in [2n-2, 2n]$. As a consequence, we have $w(n, k_1, k_2, k_3) = W(n, k_1, k_2, k_3) \leq W(x, k_1, k_2, k_3) \leq W(n+1, k_1, k_2, k_3) = w(n+1, k_1, k_2, k_3)$. Because only the variable $n$ as a parameter of the iteration is essential in this equation the above inequalities end the proof. □

To finish the current section, we recall that classical computational classes like $PF$ and $NPF$ are all subclasses of the elementary functions, and, consequently, subclasses of the primitive recursive functions.

# 4 An analytic condition for $P \subset NP$

We know that, in computability theory, the growth of functions is an important factor of its complexity. We use this approach to define two subclasses of real recursive functions. A real recursive function is said to be of exponential order if in any step of its construction, its components are bounded by the exponential function. It is said to be of subexponential order if in any step of its construction, its components are subexponentially bounded.

Concepts such as linear growth or sublinear growth, exponential growth or subexponential growth can also be applied to some arbitrary function disregarding their component functions, i.e., their inductive construction. Thus, in what follows, we distinguish between *order* and *growth*.

The formulas

$$F(s) = \int_0^\infty f(\xi)\ e^{-s\xi}\ d\xi, \quad f(\xi) = \frac{1}{2\pi i} \int_{c-i\infty}^{c+i\infty} F(s)\ e^{s\xi}\ ds$$

present the Laplace transform $F = L[f]$ and the inverse Laplace transform $f = L^{-1}[F]$, respectively. The second integral is generally carried out by contour integration.

In the rest of this paper we will consider functions of a variable $\xi$, *inter alia*, and of a complex variable $s$, *inter alia*. The function $f(..., \xi, ...)$ is called the original function and $F(.., s, ...)$ is called the image function. If the Laplace integral converges for a real $x = x_0$, i.e., the integral exists, then it exists for all $s$ with $\Re\, s > x_0$, and the image function is an analytic function of $s$ in the half-plane $\Re\, s > x_0$. The Laplace transform can be generalized to an arbitrary finite number of variables. E.g., the two dimensional Laplace transform is defined as follows:

$$F(u, v) = \int_0^\infty \int_0^\infty f(\xi, \zeta)\ e^{-u\xi - v\zeta}\ d\xi\ d\zeta.$$

We can use the Laplace transform in one dimension within many dimensions, or in more than one dimension at the same time.

With the above notions, we have a precise boundary between the subexponential order and the exponential order. **Subexponentially bounded functions** can be introduced by the following condition: for every total function $f$, $L[f](s)$ is defined along the whole positive real axis $\Re\ s > 0$. **Exponentially bounded functions**: for every total function $f$, $L[f](s)$ is defined for values of the complex variable $s$ such that $\Re\ s\ >\ x_f$, where $x_f$ depends on $f$.

Following [18], consider a real recursive function $f$ on the positive real axis such that: (i) $f$ is continuous on $[0, \infty)$ except possibly for a finite number of jump discontinuities in every finite sub-interval; (ii) there is a positive number $M$ such that $|f(\xi)| \leq Me^{k\xi}$, for all $\xi \geq 0$. Then we say that $f$ belongs to the class $L_k$. Additionally let $L = \bigcup_{k>0} L_k$.

**Proposition 24** *If $f \in L_k$ is a real recursive function, then the Laplace transform $L[f](x + iy)$ exists for $x > k$.*

**Proof.** From the condition (ii) we have $|f(\xi)|\ e^{-k\xi}\ \leq\ M$. Now we can proceed in the following way:

$$|\ \int_0^\tau\ f(\xi)\ e^{-x\xi}\ d\xi\ |\ \leq\ \int_0^\tau\ |f(\xi)|\ e^{-x\xi}\ d\xi\ \leq\ \int_0^\tau\ M\ e^{-x\xi}\ e^{k\xi}\ d\xi\ \leq\ M\ \int_0^\tau\ e^{-(x-k)\xi}\ d\xi.$$

Now to check the existence of the Laplace transform it is sufficient to take $|L[f](x+iy)|\ \leq \lim_{\tau\to\infty}\ M\ \int_0^\tau\ e^{-(x-k)\xi}\ d\xi\ = \frac{M}{x-k}$, only defined for $x > k$. □

If the Laplace transform of $f$ exists, $L[f](s)$, then $f$ is said to be of exponential order: it exists for $x = \Re\ s$ greater than some real number $x_f$. If the Laplace transform of $f$ exists, and $L[f](s)$ is defined for the all positive real axis, then $f$ has subexponential growth. It does not mean that $f$ is dominated by a polynomial since the function $\lambda x.\ x^{\log(x)}$ is subexponential but not dominated by a polynomial.

**Proposition 25** *Subexponential functions are preserved by integrals.*

**Proof.** We will prove that if $f$ is a total function of $x$, such that its Laplace transform is defined for all positive real axis, then the function $\bar{f}$ defined by

$$\bar{f}(x) = \int_0^x\ f(\xi)\ d\xi$$

is subexponential too. Let $f$ be a real valued function of $\xi$. We have

$$L[\int_0^x\ f(\xi)\ d\xi](s)\ =\ \frac{F(s)}{s}$$

whenever $L[f]\ =\ F$. This fact completes our proof, since $\bar{F}(s) = \frac{F(s)}{s}$ is defined for all the positive real axis, whenever $F$ is defined too in the same open interval. □

**Proposition 26** *Subexponential functions are preserved by differentiation, in the sense that, if a subexponential function is differentiable, then its derivative is also subexponential.*

**Proof.** We will prove that if $f$ is a total function of $x$, such that its Laplace transform is defined for the all positive real axis, then the function $\bar{f}$ defined by $\bar{f}(x) = \partial_x f(x)$ is subexponential too. We have $L[\lambda x.\ \partial_x f(x)](s) = s\,F(s) - f(0)$ whenever $L[f] = F$. This fact completes our proof, since $\lambda s.\ s\ F(s)$ is defined for all the positive real axis, whenever $F$ is defined too in the same open interval. □

Now we turn the direction of our consideration. We start from real functions and then we restrict them to the set of non negative integers.

**Definition 27** *An indexed ordered set of real numbers $\phi = \{\phi_i\}_{i\in\mathbb{N}}$ is said to be* admissible *for a function $F : \mathbb{R} \to \mathbb{R}$ if $\phi$ is a real recursive function in DAnalog and $F(\phi_i) \in \mathbb{N}$, for all $i \in \mathbb{N}$. A function $f : \mathbb{N} \to \mathbb{N}$ is said to be an* admissible restriction *of $F$ if there exists an admissible set $\{\phi_i\}_{i\in\mathbb{N}}$ such that $f(i) = F(\phi_i)$, for all $i \in \mathbb{N}$.*

We can consider functions with many variables.

**Definition 28** *A tuple of indexed ordered sets of real numbers $\{\phi_i^1\}_{i\in\mathbb{N}}, \ldots, \{\phi_i^k\}_{i\in\mathbb{N}}$ is said to be* admissible *for a function $F : \mathbb{R}^k \to \mathbb{R}$, of arity $k$, if $F(\phi^1_{i_1}, \ldots, \phi^k_{i_k}) \in \mathbb{N}$, for all $i_1, \ldots, i_k \in \mathbb{N}$, and every sequence $\{\phi^i_j\}_{j\in\mathbb{N}}$, for $i = 1, \ldots, k$, is an admissible set.* Mutatis mutandis *we get the definition of an admissible restriction of $F$.*

Not all functions have admissible restrictions, like $\lambda x.\ e^{-x}$, or just like a constant $\frac{1}{2}$. Real recursive functions that extend functions over the integers have an infinite number of admissible restrictions.

We proposed in [13] the definition of the classes *DAnalog* and *NAnalog*, which can be interpreted as classes of real recursive functions computed with *weak exponential (or quasi-polynomial) restrictions* on their values.

**Definition 29** *The class DAnalog of real recursive vector functions is inductively defined as follows:*

Primitives: *Constants* 0, 1, *and* −1, *the projections $I^i_n(x_1, \ldots, x_n) = x_i$, $1 \le i \le n$, and the functions $\theta_k(x) = x^k\Theta(x)$, $k \ge 0$, are in DAnalog:*

Composition: *if $f$ is a real recursive vector function with $n$ $k$-ary components and $g$ is a real recursive vector function with $k$ $m$-ary components, all in DAnalog, then the vector function with $n$ $m$-ary components, $1 \le i \le n$,*

$$\lambda x_1 \ldots \lambda x_m.\ f_i(g_1(x_1, \ldots, x_m), \ldots, g_k(x_1, \ldots, x_m))$$

*is in DAnalog only if all its components grow less than a weak exponential.*[2]

[2]This condition in this clause is not needed, since weak exponential bounded functions are closed for composition.

Differential recursion: *if $f$ is a real recursive vector function with $n$ $k$-ary components and $g$ is a real recursive vector function with $n$ $(k+n+1)$-ary components, both in DAnalog, then the vector function $h$ of $n$ $(k+1)$-ary components which is the solution of the Cauchy problem, $1 \leq i \leq n$,*

$$h_i(x_1, \ldots, x_k, 0) = f_i(x_1, \ldots, x_k),$$

$$\partial_y h_i(x_1, \ldots, x_k, y) = g_i(x_1, \ldots, x_k, y, h_1(x_1, \ldots, x_k, y), \ldots, h_n(x_1, \ldots, x_k, y))$$

*is in DAnalog only if all its components grow less than a weak exponential.*

Assembling and designating components: *(a) arbitrary real recursive vector functions in DAnalog can be defined by assembling scalar real recursive function components in DAnalog into a vector function; (b) if $f$ is a real recursive vector function in DAnalog, than each of its components is a real recursive scalar function in DAnalog.*

*Functions in DAnalog are said to be deterministic weak exponential.*

In the second clause we could omit the *limit of growth*, since weak exponentially bounded functions are closed for composition. But, we prefer to stress this fact because of Definitions 30 and 36.

**Definition 30** *The class NAnalog of real recursive vector functions is obtained from real recursive vector functions in DAnalog:*

Admissible bounded quantification: *if $F : \mathbb{R}^{n+1} \to \mathbb{R}$ is a scalar function in DAnalog, $\phi : \mathbb{N}^n \to \mathbb{N}$ is a polynomial, and $f : \mathbb{R}^n \to \mathbb{R}$ is a function, such that (a) $f(x_1, \ldots, x_n) \neq 0$ if and only if there exists a positive integer $k$ such that $|k| \leq \phi(||\lfloor x_1 \rfloor||, \ldots, ||\lfloor x_n \rfloor||)$ and $F(x_1, \ldots, x_n, k) \neq 0$ and (b) $f(x_1, \ldots, x_n) = y \neq 0$ if and only if there exists a positive integer $k$ such that $|k| \leq \phi(||\lfloor x_1 \rfloor||, \ldots, ||\lfloor x_n \rfloor||)$, $F(x_1, \ldots, x_n, k) = y$, and, for all such positive integer $|k| \leq \phi(||\lfloor x_1 \rfloor||, \ldots, ||\lfloor x_1 \rfloor||)$ such that $F(x_1, \ldots, x_n, k) \neq 0$, we have $F(x_1, \ldots, x_n, k) = y$, then $f$ is a scalar function in NAnalog.*

*We write $NAnalog = \exists DAnalog$. Functions in NAnalog are said to be non-deterministic weak exponential.*

We get the immediate result:

**Proposition 31** *$DAnalog \subseteq NAnalog$.*

**Example 32** *The functions $x + y$, $xy$, $x - y$, $\frac{1}{x+\epsilon}$, for all $\epsilon \in \mathbb{R}^+$, and $\frac{x}{y+\epsilon}$, for all $\epsilon \in \mathbb{R}^+$, belong to DAnalog. The weak exponential $e^{\log(x+1)^k}$, for all $k \in \mathbb{N}$, is also in DAnalog. Also* sin *and* cos *are in DAnalog.*

**Example 33** *The function exp is not in the class DAnalog. The Laplace transform of $\lambda x.\ e^x$ is $\lambda s.\ \frac{1}{s-1}$, defined for $\Re\ s > 1$. In order to belong to the class DAnalog, the transform has to be defined for $\Re\ s > 0$. The same reasoning allows the reader to verify that the function $\lambda x.\ e^{\epsilon x}$ does not belong to DAnalog no matter how small $\epsilon > 0$ is: its Laplace transform is $\lambda s.\ \frac{1}{s-\epsilon}$, defined for $\Re\ s > \epsilon$.*

Remember that functions in *DAnalog* are, strictly speaking, functions of the form $\Theta(x)f(\dots,x,\dots)$, according to Laplace transform conventions and notation, defined everywhere.

To consider some functions as subexponential, sometimes we have to make a small shift on the real variable to avoid a discontinuity at the origin. For example, the function $\lambda x.\ \frac{1}{x+\epsilon}$ is subexponential and its transform is $\lambda s.\ e^{\epsilon s}E_1(\epsilon s)$, where $E_1$ is the exponential integral of degree one, for positive $\epsilon$ as small as we want.

Using the same techniques we used in [12] to represent by means of real recursive functions the entire arithmetical and analytical hierarchies, we can define the class *NAnalog* by analytical means, without quantifiers.

Notice that, if the function of expression $f(x,y)$ is obtained in *DAnalog*, then it cannot grow faster than a weak exponential in one or in both variables. Now, if we take a univariate polynomial $\psi$ and values of $y$ such that $||\lfloor y\rfloor|| \leq \psi(||\lfloor x\rfloor||)$, then $y$ grows less than a weak exponential of $x$ and the combination of both growths, of $f$ and $y$ (now seen as function of $x$) can not grow faster than a weak exponential too. This simple explanation allows us to conclude that:

**Proposition 34** *Functions in NAnalog grow as fast as functions in DAnalog.*

In [13] we proved that every function $f \in PF$ has a canonical extension $\hat{f}$ is in *DAnalog*.

**Proposition 35** *The classes DAnalog and NAnalog are closed for integration. Moreover, DAnalog is closed for differentiation, in the sense that if a function in DAnalog is differentiable, then its derivative is in DAnalog.*

**Proof.** Consider first *DAnalog*. That this class is closed for integration derives from the facts: (a) integration is the solution of differential recursion for particular functions $f$ and $g$ (see Definition 19) and (b) the integral of a function that grows strictly less than a weak exponential, grows also strictly less than a weak exponential. Let us see what happens with differentiation. The proof is done by induction in the structure of a function $f \in DAnalog$. If $f$ is one of the basic functions, then its derivative is a basic function. If $f$ of arity $n$ arises from composition of functions $g$ of arity $k$ and $h_1$, ...,$h_n$ of arity $n$, then, by induction hypothesis, the derivatives of $g$, $h_1$, ..., $h_n$ are in *DAnalog* (which is closed under composition). We have then

$$\partial_x f(\dots,x,\dots) = \sum_{i=1}^{k} \partial_{y_i} g(y_1,\dots,y_k)_{|y_i=h_i(\dots,x,\dots)} \partial_x h_i(\dots,x,\dots).$$

We conclude that the derivative of $f$ is in *DAnalog*, since it can not grow above some weak exponential.

Finally, if $f$ is solution of the differential recursion scheme

$$f(\dots,x,\dots,0) = h(\dots,x,\dots),$$

$$\partial_y f(\dots, x, \dots, y) = g(\dots, x, \dots, y, f(\dots, x, \dots, y)),$$

then it follows that, by induction hypothesis, $\partial_x h(\dots, x, \dots)$ and $\partial_x g(\dots, x, \dots, y, z)$ are in $DAnalog$. We write the following differential scheme

$$\partial_x f(\dots, x, \dots, 0) = \partial_x h(\dots, x, \dots),$$

$$\partial_y \partial_x f(\dots, x, \dots, y) = \partial_x g(\dots, x, \dots, y, f(\dots, x, \dots, y))$$

$$= (\partial_x g(\dots, x, y, z))_{|z=f(\dots,x,\dots,y)} + (\partial_z g(\dots, x, y, z)))_{|z=f(\dots,x,\dots,y)} \partial_x f(\dots, x, \dots, y),$$

that together with the previous differential scheme for $f$ provide the solutions of both $f$ and and its derivative.

That $NAnalog$ is closed for integration follows exactly as for $DAnalog$. □

Now we propose the definition of the classe $fAP$ (the $f$ stands for *feasible*), which can be interpreted a the class of real recursive functions computed with *quasi-polynomial restrictions* on their values and times of computation. Of course $fAP$ — *analog quasi-polynomial time* in this context is just rhetoric, but as we will see these subclasses of real recursive functions arise in such natural way that they are not superseded by their classical counterparts.

**Definition 36** *The class of real recursive functions designated by $fAP$ is defined as $DAnalog$, but substituting the weak exponential order by subexponential order.*

*Functions in $fAP$ are said to be deterministic subexponential.*

We see that our $fAP$ indeed captures the meaning of Odifreddi words from Section VIII of [15] including all stepwise subexponential functions, providing a quite meaningful computational class. In classical terms, this class is not easy to capture or characterize, since there exists not a operational method to define it. We can only characterize the subexponential functions in the classical framework by means of quantifiers: for every total function $f$, $f$ is subexponential if, for all $\epsilon > 0$, there exists a positive integer $M$, such that $f(x) < Me^{\epsilon x}$.

Let us conclude with the definition of the two following classes:

**Definition 37** *Let $DAnalog^r$ and $NAnalog^r$ be the restrictions of the classes $DAnalog$ and $NAnalog$ to functions that in any step of their construction have admissible restrictions in $PF$ and $NPF$, respectively.*

We know from [13], that all functions in $PF$ or in $NPF$ have extensions in $DAnalog^r$ and $NAnalog^r$, respectively. In [13] we proved that (let us recall here that $NPF \subseteq PF$ iff $NP \subseteq P$):

**Proposition 38** *If a function has an admissible restriction in any step of its construction, then it belongs to $DAnalog$.*

**Proof.**[3] Suppose that a function $f$ of arity one in these conditions is not in $DAnalog$. Then we can find, in some step of the construction a component $g$

[3]This proof can be given by structural induction. Herein we provide this rather different proof.

such that, for all $k \in \mathbb{N}$, $g(x) \geq 2^{\log(x)^k}$. The function $g$ has an admissible restriction which is not quasi-polynomially bounded, contrarily to our hypothesis. Thus $f$ is in $DAnalog$. □

The following statement although similar with final result in our previous paper [13], is more strongly based in a clear mathematical formulation (avoiding Proposition 31 of [13] that, it seems, it is not well-founded [4]).

**Proposition 39** *If $NPF \subseteq PF$, then $NAnalog^r \subseteq DAnalog^r$.*

**Proof.**[5] If $f$ is a function in $NAnalog$, then all their admissible restrictions along its construction are nondeterministic quasi-polynomially bounded (i.e., they are in $NPF$), and then, by hypothesis, they are deterministic quasi-polynomially bounded (i.e., they are in $PF$). We end the proof applying Proposition 38. □

Since all functions in $DAnalog^r$ are built from components $f$ satisfying (a) a differential equation that can be seen as linear and (b) a growing condition, then the differential equation itself can be solved by Laplace transforms. The above considerations justify the following interpretation of Proposition 39.

**The problem whether $P$ is a proper subclass of $NP$ can be transformed into the problem of a proof $DAnalog^r \neq NAnalog^r$, which will be based on some sets of real functions with Laplace transforms**.

# 5 Why is $fAP$ elegant?

In this section we want to strongly motivate our concept of feasible functions, computable over the real numbers, i.e. the functions of $fAP$. To do such an exercise we will be helped by a book on analog computation from the sixties (see [1]). Taking the examples throughout the book we can discuss the borderline between feasible and non feasible analog computation.

We define first a proper subclass $LIN(\mathbb{R})$ of $REC(\mathbb{R})$, by restricting differential recursion to linear differential recursion (see [7]).

**Definition 40** *$LIN(\mathbb{R})$ The class $LIN(\mathbb{R})$ of real recursive vectors is generated from the real recursive scalars $0,\ 1,\ -1,\ \pi$, primitive functions $\theta_k(x) = x^k\Theta(x)$, for $k \geq 0$, and projections $I^i_n(x_1, \ldots, x_n) = x_i,\ 1 \leq i \leq n,\ n > 0$, by the following operators: assembling and designating components, composition, and*

Linear differential recursion: *if $f$ is a real recursive vector with $n$ $m$-ary components and $g$ is a real recursive matrix with $m \times m$ $(n+1)$-ary components, then the vector $h$ of $n$ $m$-ary components which is the solution of the Cauchy problem, $1 \leq i \leq n$, $h_i(x_1, \ldots, x_m, 0) = f_i(x_1, \ldots, x_m)$, $\partial_y h_i(x_1, \ldots, x_m, y) = \sum_{j=1}^{m} g_{ij}(x_1, \ldots, x_m, y)\ h_j(x_1, \ldots, x_m, y)$ is real recursive.*

[4] Although it is not obvious, and more work has to be done with regard to cleaner and simpler formulation of such an analytic conditon — indeed, intended to shed new light on $P \subset NP$ conjecture.

[5] *idem, like in the last proof.*

Note that linear integration can only solve differential equations of the form $\partial h = gh$, where the right-hand side is linear in $h$, rather than the arbitrary dependence $\partial h = g(h)$ of which $REC(\mathbb{R})$ is capable. Secondly, we can expand our set of variables, and so solve non-homogeneous linear differential equations of the form $\partial h = gh+b$. In [7] the following results can be found: $LIN(\mathbb{R})$ contains, e.g., sin, cos, $\lambda x.\ e^x$, the rational numbers, and real recursive extensions of successor, addition, and cut-off subtraction; moreover all functions in $LIN(\mathbb{R})$ are continuous and total.

It is proved in [7] that there exist bounds on the growth of functions in $LIN(\mathbb{R})$, namely:

**Proposition 41** *Let $h : \mathbb{R}^m \to \mathbb{R}$ be a function in $LIN(\mathbb{R})$. Then there is a constant $d$ such that, up to multiplicative constants, for all $(x_1, \ldots, x_m) \in \mathbb{R}^m$,*

$$\|h(x_1, \ldots, x_m)\| \leq e^{[d]}(\|(x_1, \ldots, x_m)\|),$$

$$\|\partial_{x_i} h(x_1, \ldots, x_m)\| \leq e^{[d]}(\|(x_1, \ldots, x_m)\|),$$

*where by $e^{[n]}$ we mean the iterated exponential: $e^{[0]}$ is just the identity function and $e^{[n+1]}(x) = e^{e^{[n]}(x)}$.*

Proposition 41 establish the same kind of bounds as for Kalmar's elementary functions. But the relation between these two classes can be shown to be much tighter: namely, all functions in $LIN(\mathbb{R})$ can be approximated (in the sense of Grzegorczyk, e.g., like in [19]) by elementary functions, and all real recursive extensions of elementary functions are contained in $LIN(\mathbb{R})$. Proofs of these statements can be found in [7].

When the matrix $g$, in the definition of linear differential recursion, is made of constant real recursive entries, then the last proposition takes a very particular and interesting form. Namely, if $f$ is such a function from $LIN(\mathbb{R})$, that in any step of application of linear differential recursion the matrix $g$ is constant, then up to multiplicative constants, we have for all $(x_1, \ldots, x_m) \in \mathbb{R}^m$,

$$\|h(x_1, \ldots, x_m)\| \leq e^{\|(x_1, \ldots, x_m)\|}, \quad \|\partial_{x_i} h(x_1, \ldots, x_m)\| \leq e^{\|(x_1, \ldots, x_m)\|}.$$

It is not known, up to our knowledge, if such constraints on the matrix are enough to describe the full space of functions that have stepwisely a Laplace transform.

The fact about $LIN(\mathbb{R})$ is that it contains extensions of $PF$ and $NPF$. Also important for us is the connection between linear differential equations and the Laplace transform. Using in this context the Laplace transform, operations of differentiation and integration can be replaced with algebra.

Now, consider the linear differential recursion scheme $\partial h = gh + b$. In the case we have the null matrix $g$, the scheme reduces to

$$\partial_y h(x_1, \ldots, x_m, y) = b(x_1, \ldots, x_m, y),$$

and we can conclude, by virtue of the fact that subexponential functions are closed under integration, that if $b$ is subexponential, then $h$ is subexponential

too. This constitute the subclass of $LIN(\mathbb{R})$ which is built up just by simple integration. We consider in what follows the integral form of linear differential recursion:

$$h(x_1,\dots,x_m,y) = h(x_1,\dots,x_m,0) +$$

$$\int_0^y g(x_1,\dots,x_m,t)\ h(x_1,\dots,x_m,t)\ dt + \int_0^y b(x_1,\dots,x_m,t)\ dt.$$

Let $\gamma$ be such that $\gamma(x_1,\dots,x_m,t-y) = g(x_1,\dots,x_m,y)$. Such function is in $LIN(\mathbb{R})$ because it can be obtained by composition $\gamma(x_1,\dots,x_m,y) = g(x_1,\dots,x_m,t-y)$.

The integral form becomes now

$$h(x_1,\dots,x_m,y) = h(x_1,\dots,x_m,0) +$$

$$+ \int_0^y \gamma(x_1,\dots,x_m,y-t)\ h(x_1,\dots,x_m,t)\ dt + \int_0^y b(x_1,\dots,x_m,t)\ dt.$$

We can prove the following theorem:

**Proposition 42** *$fAP$ ($\bigcap LIN(\mathbb{R})$) consists of the functions which in any step of construction satisfy $L[\gamma] \leq 1$ in the whole positive real axis.*

**Proof.** Applying the Laplace transform to the integral form, we get

$$H(x_1,\dots,x_m,s) =$$

$$= \frac{h(x_1,\dots,x_m,0)}{s} + H(x_1,\dots,x_m,s)\ L[\gamma(x_1,\dots,x_m,y)](s) + \frac{B(x_1,\dots,x_m,s)}{s},$$

from where it follows that

$$H(x_1,\dots,x_m,s) = \frac{h(x_1,\dots,x_m,0) + \frac{B(x_1,\dots,x_m,s)}{s}}{s(1 - L[\gamma(x_1,\dots,x_m,y)](s))}.$$

We conclude that the solution is in $fAP$ if and only if $L[\gamma(x_1,\dots,x_m,y)](s) \leq 1$, for all $\Re\ s > 0$, since from the inductive point a view, functions $\gamma$ and $b$ are already subexponential. □

**Example 43** *Considering the scalar case $h(0) = 1$, $\gamma(y) = 1$, and $b(y) = 0$, we obtain $L[\gamma(y)] = \frac{1}{s}$ that do not satisfy the condition of our statement (e.g., in the open interval $(0,1)$). In fact, in this case, we have $H(s) = \frac{1}{s(1-\frac{1}{s})} = \frac{1}{s-1}$. Applying the Bromwich contour we find $h(x) = e^x$, as we expected.*

The integral version of the linear differential recursion scheme is a generalized form of the *Volterra integral equation* that can be written in the standard form

$$y(t) = f(t) + \int_0^t y(\tau)\ K(t-\tau)\ d\tau,$$

for $t > 0$. The function K is called the *kernel* of the equation (cf. [9]). This result is quite interesting because an ordinary differential equation may be transformed into an integral equation. For example, if $y(x)$ satisfies the $n$-th order ordinary differential equation $y^{(n)}(x) = f(x) + \sum_{j=1}^{n} C_j(x)\ y^{(j-1)}(x)$ and $u(x) = y^{(n)}(x)$, then $u(x)$ satisfies the integral equation $u(x) = \phi(x) + \int_a^x k(x,t)\ u(t)\ dt$, with $k(x,t) = \sum_{j=1}^{n} C_j(x)\ \frac{(t-x)^{j-1}}{(j-1)!}$, where $\phi(x)$ is $f(x)$ plus a polynomial in $(x-a)$ generated by the initial conditions.

By *linear differential equation* (see, e.g., [20]) we mean a differential equation where the dependent variable appears only with exponent 0 or 1. All differential equations in the sense of this definition are in $LIN(\mathbb{R})$. (As a counterexample, the equation $y\ \partial_x y\ =\ 1$ is not linear.) Then we can state the following:

**Proposition 44** *A linear differential equation has a solution in* $fAP$ $(\bigcap LIN(\mathbb{R}))$ *whenever* $\phi$ *is subexponential and the kernel* $k$ *satisfy the Laplacian conditioning of Proposition 42.*

**Proof.** Proposition 42 offers all ingredients of the proof, applying the Laplace transform to the Volterra integral equation

$$u(x) = \phi(x) + \int_a^x k(x,t)\ u(t)\ dt.$$

Since $u(x) = y^{(n)}(x)$, to obtain the Laplace transform of $y$ we have to solve the algebraic equation $U(s) = s^n\ Y(s) - s^{n-1}\ f(0) - s^{n-2}\ y'(0) - ... - y^{(n-1)}(0)$. □

One good idea seems to be defining the matrix of the linear differential recursion scheme in order to, with the help of the Laplace transform, characterize the subexponential world, and, within it, different classes, possibly including $P$, $NP$, and lower complexity classes. From the physical point a view, the last example is an example of a dissipative system, with a dumping term. A class of Hamiltonian systems and a class of dissipative systems is captured by our subexponential world containing extensions of $DAnalog$ and $fAP$.

Let us stress that many physical systems are in $fAP$.

*RC circuits* are in $fAP$, satisfying the differential equation

$$RC\ \partial_t \upsilon + \upsilon = V(t),$$

where the voltage source is $V(t)$ and the initial condition is $\upsilon(0) = \upsilon_0$. Solving by Laplace transform we find $L[\upsilon] = \frac{\upsilon_0}{s+\frac{1}{RC}} + \frac{1}{RC(s+\frac{1}{RC})}\ L[V(t)]$. Thus, if $V$ is subexponential, then $\upsilon$ is subexponential.

*RLC circuits* are in $fAP$, satisfying the system of differential equations

$$R\ i + \upsilon_C + \upsilon_L = V(t),$$

$$C\ \partial_t \upsilon_C = i,$$

$$L\ \partial_t i = \upsilon_L,$$

where the voltage source is $V(t)$, the voltage across the capacitor is $\upsilon_C$, the voltage across the inductor is $\upsilon_L$, and the initial conditions are $\upsilon_C(0) = \upsilon_{C,0}$

and $\upsilon_L(0) = \upsilon_{L,0}$. Also we let $R$ denote the resistance, $C$ the capacitance, and $L$ the inductance of the associated components of the circuit. Solving by Laplace transform we find that, if $V$ is subexponential, then $\upsilon_C$ and $\upsilon_L$ are subexponential.

*Harmonic oscillators* are in $fAP$, satisfying differential equations of the form

$$\partial_t \partial_t \upsilon + p\ \partial_t \upsilon + q\ \upsilon = f(t),$$

where $f$ is a *forcing function*. E.g., taken from [3] the harmonic oscillator described by $\partial_t \partial_t \upsilon + 4\ \partial_t \upsilon = 3\cos(t)$, has transform $L[\upsilon] = \frac{3s}{(s^2+4)(s^2+1)}$, and Bromwich contour $\upsilon(t) = -\cos(2t) + \cos(t)$. Of course these physical systems are in $fAP$, whenever the forcing function $f$ is.

# 6 Final Remark

If this paper had been written just to have a focus on the $P \neq NP$, then we would not need at all the concept of generalized solution introduced in Section 3. It would be enough to work with linear integration, concept introduced in Section 5, because the class of elementary functions discussed in that section contains both $PF$ and $NPF$. Linear integration has a unique solution. However, defining $REC_R(R)$ exactly as we do in Section 3 makes our framework mathematically more interesting. But the reader should be aware that a simpler framework, that of linear integration, is suitable for the purpose of this paper.

# 7 Acknowledgement

We thank to Kerry Ojakian for point us a few potential problems in the formalization of [13], situations that we don't know yet how to solve. These problems were avoided in the formulation of this paper.

# References

[1] J. Robert Ashley. *Introduction to Analog Computation*, John Wiley and Sons, Inc., 1963.

[2] José L. Balcázar, Josep Díaz, and Joaquim Gabarró. *Structural Complexity I*, Springer-Verlag, Second Edition, 1995.

[3] Paul Blanchard, Robert L. Devaney, and Glen R. Hall. *Differential Equations*, Brooks/Cole Publishing Company, 1998.

[4] Michael S. Branicky. Universal computation and other capabilities of hybrid and continuous dynamical systems. *Theoretical Computer Science*, 138(1):67-100, 1995.

[5] Manuel L. Campagnolo. Computational complexity of real valued recursive functions and analog circuits, PhD dissertation, Universidade Técnica de Lisboa, 2001.

[6] Manuel L. Campagnolo, Cristopher Moore, and José Félix Costa. Iteration, inequalities, and differentiability in analog computers. *Journal of Complexity*, 16(4):642-660, 2000.

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

[8] Daniel Graça and José Félix Costa. Analog computers and recursive functions over the reals. *Journal of Complexity*, 19(5): 644-664, 2003.

[9] A. C. King, J. Billingham, and S. R. Otto. *Differential Equations, Linear, Nonlinear, Ordinary, Partial*, Cambridge University Press, 2003.

[10] Cristopher Moore. Recursion theory on the reals and continuous-time computation. *Theoretical Computer Science,* 162: 23-44, 1996.

[11] Jerzy Mycka. $\mu$-recursion and infinite limits. *Theoretical Computer Science*, 302: 123-133, 2003.

[12] Jerzy Mycka and José Félix Costa. Real recursive functions and their hierarchy. *Journal of Complexity*, 20(6): 835-857, Elsevier, 2004.

[13] J. Mycka and J. F. Costa. The $P \neq NP$ conjecture in the context of real and complex analysis. *Journal of Complexity*, 22 (2): 287-303, 2006.

[14] Piergiorgio Odifreddi. *Classical Recursion Theory I*, Elsevier, 1992.

[15] Piergiorgio Odifreddi. *Classical Recursion Theory II*, Elsevier, 1999.

[16] D. Richardson. *Journal of Symbolic Logic*, 33:514, 1968.

[17] Claude Shannon. Mathematical theory of the differential analyzer. *J. Math. Phys. MIT*, 20:337-354, 1941.

[18] Anders Vretblad. *Fourier Analysis and Its Applications*, Graduate Texts in Mathematics 223, Springer-Verlag, 2003.

[19] Klaus Weihrauch. *Computable Analysis, An Introduction*, Texts in Theoretical Computer Science, Springer-Verlag, 2000.

[20] Daniel Zwillinger. *Handbook of Differential Equations*, Academic Press, Third Edition, 1989, 1998.