

Propiedades básicas de los números computables

Carlos M. Parra

Johany A. Suárez

Recibido Mar. 31, 2006

Aceptado Sep. 5, 2006

Abstract

Motivated by Algorithmic Information Theory, we study the field $\mathbb{C}_c$ of the complex computable numbers. We also present some non trivial examples of computable and noncomputable real numbers and propose a definition of a partial computable function defined in $\mathbb{C}_c^n$. We develop the basic notions of convergence in the field $\mathbb{R}_c$ of the real computable numbers and show that every analytic function whose series representation has computable coefficients is a computable function when restricted to any closed disk in its domain. Finally, we prove that $\mathbb{R}_c$ is a real closed field and also that $\mathbb{C}_c$ is an algebraic closed field containing the algebraic numbers.

Keywords: Real numbers, Computable numbers, Constructive analysis.

AMSC(2000): Primary: 03D80, Secondary: 03F60, 26E40

Resumen

Motivados por la Teoría Algorítmica de la Información, estudiamos el campo $\mathbb{C}_c$ de los números complejos computables. Presentamos ejemplos no triviales de números reales tanto computables como no computables y proponemos una definición de función parcial computable definida en $\mathbb{C}_c^n$. Desarrollamos la convergencia en el campo $\mathbb{R}_c$ de los reales computables y mostramos que toda función analítica, cuya expansión en serie de potencias tenga coeficientes computables, es una función computable cuando se restringe a un disco cerrado de su dominio. Por último, probamos que $\mathbb{R}_c$ es un campo real cerrado y que $\mathbb{C}_c$ es un campo algebraicamente cerrado que contiene a los números algebraicos.

Palabras y frases claves: Números reales, Números computables, Análisis constructivo.

1 Introducción

Un resultado bastante mencionado en la literatura sobre Aleatoriedad e Información es que todo número real aleatorio es trascendente. Esto se debe a que todo número algebraico es computable, un hecho que suele citarse sin prueba. En lo que sigue, ofrecemos un desarrollo, en gran medida autocontenido, de las propiedades básicas de los números y las funciones computables, incluyendo una demostración de que los números computables forman un campo algebraicamente cerrado que incluye a los números algebraicos.

En la lógica matemática y el análisis constructivo, los *números computables*, también conocidos como *números recursivos*, son aquellos que se pueden computar por un algoritmo. Estos números fueron introducidos por Alan Turing en su artículo pionero de 1936 y definidos independientemente por Post y Church (ver [5]). Desde entonces se han ofrecido varias definiciones equivalentes. La exposición de los temas se basa en gran medida en [8]. Por conveniencia, seguiremos el enfoque desarrollado en [1] y [2], de los cuales modificamos algunas pruebas. Supondremos que el lector está familiarizado con

las nociones básicas y resultados fundamentales de la teoría de la computabilidad así como con la *Tesis de Church-Turing*, tal como se expone en textos como [1] ó [4]. En lo que sigue adoptaremos la práctica usual de fijar una enumeración calculable $\{M_e\}$, $e \in \mathbb{N}$, de las máquinas de Turing y denotar por $\varphi_e^{(m)}$ la función *recursiva parcial* de m variables computada por la máquina de Turing M_e . En ciertas aplicaciones, necesitaremos el conocido teorema *s-m-n* de Kleene.

Teorema 1.1. *Para todo $m, n, k \geq 1$, existe una función recursiva total $s_n^m : \mathbb{N} \times \mathbb{N}^m \rightarrow \mathbb{N}$ tal que para todo $e \in \mathbb{N}$ si $\varphi_e^{(n+m)} : \mathbb{N}^{m+n} \rightarrow \mathbb{N}^k$ es recursiva parcial, entonces para todo $\mathbf{v} \in \mathbb{N}^m$ se cumple que*

$$\varphi_{s(e, \mathbf{v})}^{(n)}(\mathbf{u}) = \varphi_e^{(n+m)}(\mathbf{v}, \mathbf{u}), \quad \text{para todo } \mathbf{u} \in \mathbb{N}^n.$$

Recordemos que $A \subseteq \mathbb{N}^m$ es *recursivamente enumerable* si $A = \text{dom}(\varphi)$, con φ recursiva parcial y que A es *recursivo* si tanto A como A^c son recursivamente enumerables (o equivalentemente si su correspondiente función característica $\chi_A : \mathbb{N}^m \rightarrow \{0, 1\}$ es recursiva). El ejemplo más célebre de un conjunto recursivamente enumerable que *no* es recursivo es el *conjunto de parada*

$$\mathbf{K} = \{n \mid n \in \text{dom}(\varphi_n)\},$$

cuya importancia es vital en la teoría de la recursión.

2 Definiciones básicas y códigos

Fijemos dos biyecciones computables $\sigma : \mathbb{N}^2 \rightarrow \mathbb{N}$ y $\nu : \mathbb{N} \rightarrow \mathbb{Q}$ cuyas inversas también sean computables. Diremos que la función $\psi : \mathbb{N}^m \rightarrow \mathbb{Q}$ es *recursiva parcial* si la función $\varphi^{(m)} = \nu^{-1} \circ \psi : \mathbb{N}^m \rightarrow \mathbb{N}$ es recursiva parcial. En general, $\psi : \mathbb{N}^m \rightarrow \mathbb{Q}^k$, con $\psi = (\psi_1, \dots, \psi_k)$, es *recursiva parcial* si cada una de sus componentes $\psi_i : \mathbb{N}^m \rightarrow \mathbb{Q}$ es recursiva parcial. En particular, una función recursiva total $f : \mathbb{N} \rightarrow \mathbb{Q}$ se denomina una *sucesión computable de números racionales* y se denotará por $(q_n)_{n=1}^\infty$, donde $q_n = f(n)$. Análogamente al caso natural, $A \subseteq \mathbb{Q}$ es *recursivo* si su función característica $\chi_A : \mathbb{Q} \rightarrow \{0, 1\}$ es recursiva.

Definición 2.1. *Decimos que $\alpha \in \mathbb{R}$ es computable si existe una sucesión computable $(q_n)_{n=1}^\infty$ de racionales tal que*

$$|\alpha - q_n| \leq \frac{1}{2^n}, \quad \text{para todo } n.$$

La función recursiva $f : \mathbb{N} \rightarrow \mathbb{Q}$ que satisface $f(n) = q_n$ se denomina *función de aproximación para α* . Diremos que $z \in \mathbb{C}$ es *computable* si $\text{Re } z$ é $\text{Im } z$ son reales computables. Por comodidad, introducimos la siguiente notación:

$$\mathbb{R}_c = \{\alpha \in \mathbb{R} \mid \alpha \text{ es computable}\}, \quad \mathbb{C}_c = \{a + bi \mid a, b \in \mathbb{R}_c\}$$

y

$$\mathbb{A} = \{z \in \mathbb{C} \mid z \text{ es algebraico}\}.$$

Si identificamos $\mathbb{Q} + i\mathbb{Q}$ con $\mathbb{Q}^2$, se puede ver fácilmente que $z \in \mathbb{C}_c$ si, y sólo si, existe una función total recursiva $g : \mathbb{N} \rightarrow \mathbb{Q}^2$ tal que $g(n) = r_n + is_n$ y se satisface que

$$|z - (r_n + is_n)| \leq \frac{1}{2^n}, \quad \text{para todo } n.$$

Análogamente al caso real, denominaremos $g : \mathbb{N} \rightarrow \mathbb{Q}^2$ una *función de aproximación para z* .

Informalmente, describir un número computable z se reduce a determinar un índice e tal que la máquina de Turing M_e genera aproximaciones para z en el sentido de las anteriores definiciones. En general, es imposible operar directamente con los números computables. En lugar de ello, se elige un *método de codificación computable* adecuado y se opera con los índices correspondientes.

Necesitamos describir con detalle una forma adecuada de codificar k -tuplas de números computables. Por brevedad, denotemos por

$$\mathfrak{N} = \{\mathbb{N}, \mathbb{Q}, \mathbb{R}_c, \mathbb{C}_c\}$$

la familia de sistemas numéricos computables.

Definición 2.2. *Supongamos que $A \in \mathfrak{N}$. Un código para $z \in A$ es un número natural e que se define por casos:*

- (a) *si $A = \mathbb{N}$, entonces $e = z$.*
- (b) *si $A = \mathbb{Q}$, entonces $e = \nu^{-1}(z)$.*
- (c) *si $A = \mathbb{R}_c$, entonces e es el índice de una función de aproximación $\varphi_e : \mathbb{N} \rightarrow \mathbb{Q}$ para z .*
- (d) *si $A = \mathbb{C}_c$, entonces $e = \sigma(e_1, e_2)$ donde e_1, e_2 son códigos para $\operatorname{Re} z$ é $\operatorname{Im} z$ de acuerdo a los casos (a) – (c).*

Por ejemplo, un código para $z = 2 + \pi i$ sería un natural $n = \sigma(2, m)$, donde $\varphi_m : \mathbb{N} \rightarrow \mathbb{Q}$ es una función de aproximación para π .

En general, sean $A_1, \dots, A_k \in \mathfrak{N}$. Un código para $\mathbf{z} = (z_1, \dots, z_k) \in A_1 \times \dots \times A_k$ es una k -tupla $\mathbf{e} = (e_1, \dots, e_k) \in \mathbb{N}^k$ donde e_i es un código para $z_i \in A_i$ en el sentido de la definición 2.2, para todo $i = 1, \dots, k$.

A primera vista, esta noción de código puede parecer algo artificial, pero es consistente con la práctica usual en el análisis constructivo.

Lema 2.3. *Para todo $A, A' \in \mathfrak{N}$ existe una función parcial recursiva $\gamma : \mathbb{N} \rightarrow \mathbb{N}$ tal que para todo $z \in A \cap A'$, si e es un código para $z \in A$, entonces $\gamma(e)$ es un código para $z \in A'$.*

Demostración. Consideremos algunos de los casos más relevantes. – Para $A = \mathbb{R}_c$ y $A' = \mathbb{N}$, dado $e \in \mathbb{N}$ definimos $\gamma(e)$ como el menor natural k tal que

$$|k - \varphi_e(2)| < \frac{1}{2}.$$

Ahora, supongamos que e es un código para $z \in A$. Si $z \in A \cap A' = \mathbb{N}$, entonces k existe y, en dicho caso, $k = z$. Por tanto, $\gamma(e) = k$ es un código para $z \in A'$. – Para $A = \mathbb{R}_c$ y $A' = \mathbb{C}_c$, dado $e \in \mathbb{N}$ definimos

$$\gamma(e) = \sigma(e, 0).$$

Ahora, si e es un código para $z \in A$ y $z \in A \cap A' = \mathbb{R}_c$, entonces $\gamma(e)$ es un código para $z \in A'$. – Para $A = \mathbb{C}_c$ y $A' = \mathbb{Q}$, dado $e \in \mathbb{N}$ computamos el único $(e_1, e_2) \in \mathbb{N}^2$ tal que $\sigma(e_1, e_2) = e$ y definimos $\gamma(e) = e_1$. Ahora, si e es un código para $z \in A$, entonces e_1 es un código para $\text{Re } z$. Si además $z \in A'$, entonces $z = \text{Re } z$ y $\gamma(e) = e_1$ es un código para $z \in A'$. – Para $A = \mathbb{N}$ y $A' = \mathbb{R}_c$, primero consideramos una función total recursiva $\psi : \mathbb{N}^2 \rightarrow \mathbb{Q}$ dada por

$$\psi(e, n) = e.$$

Por el teorema *s-m-n*, existe una función total recursiva $\gamma : \mathbb{N} \rightarrow \mathbb{N}$ tal que para todo $n : \varphi_{\gamma(e)} = \psi(e, \cdot)$. Si e es un código para $z \in A$, entonces $z = e$. Por tanto, $\varphi_{\gamma(e)}$ es una función de aproximación para z . Esto es, $\gamma(e)$ es un código para $z \in A'$. Los restantes casos se pueden obtener de forma análoga. $\square$

Como consecuencia inmediata se obtiene el siguiente corolario.

Corolario 2.4. *Para todo $A_i, A'_i \in \mathfrak{N}$, con $1 \leq i \leq k$, existe una función parcial recursiva $\gamma : \mathbb{N}^k \rightarrow \mathbb{N}^k$ tal que para todo $\mathbf{z} \in \prod A'_i \cap \prod A_i$, si $\mathbf{e}$ es un código para $\mathbf{z} \in \prod A'_i$, entonces $\gamma(\mathbf{e})$ es un código para $\mathbf{z} \in \prod A_i$.*

Ahora, proponemos una generalización de la noción de función parcial recursiva.

Definición 2.5. *Sean $A_1, \dots, A_k, B_1, \dots, B_j \in \mathfrak{N}$. Decimos que una función parcial $F : A_1 \times \dots \times A_k \rightarrow B_1 \times \dots \times B_j$ es computable si existe una función parcial recursiva $\theta : \mathbb{N}^k \rightarrow \mathbb{N}^j$ tal que para todo $\mathbf{z} \in \text{dom}(F)$: si $\mathbf{e}$ es un código para $\mathbf{z}$, entonces*

$$\mathbf{e} \in \text{dom}(\theta) \quad \text{y} \quad \theta(\mathbf{e}) \text{ es un código para } F(\mathbf{z}).$$

En la definición anterior, surge una ambigüedad respecto a la computabilidad de F si su dominio se considera contenido en un conjunto de salida diferente $A'_1 \times \dots \times A'_k$ (similarmente si modificamos su conjunto de llegada). La ambigüedad consiste en que se pueden generar dos funciones parciales $F, \hat{F}$ con el mismo gráfico y no es obvio que una sea computable si y sólo si la otra lo es. Esta aparente inconsistencia de nuestra definición se resuelve a continuación.

Lema 2.6. Sean $A_i, A'_i, B \in \mathfrak{N}$, con $1 \leq i \leq k$ y $F : A_1 \times \cdots \times A_k \rightarrow B$ una función parcial. Si $\text{dom}(F) \subseteq A'_1 \times \cdots \times A'_k$, definimos $\widehat{F} : A'_1 \times \cdots \times A'_k \rightarrow B$ por

$$\widehat{F}(\mathbf{z}) = \begin{cases} F(\mathbf{z}), & \text{si } \mathbf{z} \in \text{dom}(F), \\ \text{indefinido}, & \text{en caso contrario.} \end{cases}$$

Entonces F es computable sii $\widehat{F}$ es computable.

Demostración. Basta mostrar que si F es computable, entonces $\widehat{F}$ es computable. Luego, supongamos que existe una función parcial recursiva $\theta : \mathbb{N}^k \rightarrow \mathbb{N}$ con las propiedades de la definición 2.5. Basta tomar la función parcial recursiva $\widehat{\theta} = \theta \circ \gamma$, donde $\gamma : \mathbb{N}^k \rightarrow \mathbb{N}^k$ es la función dada por el corolario 2.4. Por tanto, $\widehat{F}$ es computable. $\square$

De la anterior discusión técnica, podemos concluir que *intuitivamente una función computable está determinada por un procedimiento efectivo que toma códigos apropiados para elementos de su dominio y los envía en códigos para las correspondientes imágenes.*

Las siguientes propiedades son ahora inmediatas.

Proposición 2.7. (a) Una función $F : \mathbb{C}_c^k \rightarrow \mathbb{C}^m$ es computable sii cada componente $F_i : \mathbb{C}_c^k \rightarrow \mathbb{C}$ es computable.

(b) Una función $F : \mathbb{C}_c^k \rightarrow \mathbb{C}$ es computable sii $\text{Re}(F) : \mathbb{C}_c^k \rightarrow \mathbb{R}$ y $\text{Im}(F) : \mathbb{C}_c^k \rightarrow \mathbb{R}$ son computables.

(c) Si $F : \mathbb{C}_c^k \rightarrow \mathbb{C}_c^m$ y $G : \mathbb{C}_c^m \rightarrow \mathbb{C}_c^r$ son funciones parciales computables, entonces, $G \circ F$ es computable.

En analogía con la familia de funciones primitivas recursivas, introducimos las siguientes operaciones entre funciones.

Definición 2.8. (a) *Sustitución:*

$$f(w_1, \dots, w_n) = g(h_1(w_1, \dots, w_n), \dots, h_m(w_1, \dots, w_n)).$$

Decimos que $f : \mathbb{C}^n \rightarrow \mathbb{C}$ se obtuvo por sustitución de las funciones $g : \mathbb{C}^m \rightarrow \mathbb{C}, h_1 : \mathbb{C}^n \rightarrow \mathbb{C}, \dots, h_m : \mathbb{C}^n \rightarrow \mathbb{C}$.

(b) *Recursión:*

$$\begin{aligned} f(w_1, \dots, w_k, 0) &= g(w_1, \dots, w_k) \\ f(w_1, \dots, w_k, n+1) &= h(w_1, \dots, w_k, n, f(w_1, \dots, w_k, n)) \end{aligned}$$

Decimos que $f : \mathbb{C}^k \times \mathbb{N} \rightarrow \mathbb{C}$ se obtuvo por recursión de $g : \mathbb{C}^k \rightarrow \mathbb{C}$ y $h : \mathbb{C}^k \times \mathbb{N} \times \mathbb{C} \rightarrow \mathbb{C}$.

Teorema 2.9. *La familia de funciones computables es cerrada bajo las operaciones de sustitución y recursión.*

Demostración. Para la sustitución, basta aplicar la proposición 2.7 (a) y (c). Ahora para la clausura bajo recursión, dadas las funciones computables $g : \mathbb{C}_c^k \rightarrow \mathbb{C}_c$ y $h : \mathbb{C}_c^k \times \mathbb{N} \times \mathbb{C}_c \rightarrow \mathbb{C}_c$, el siguiente procedimiento computa $f(w_1, \dots, w_k, n)$:

1. Haga $w := g(w_1, \dots, w_k)$.
2. si $n = 0$, imprima w y pare
3. en caso contrario
 para $i = 1$ hasta n haga
 $w := h(w_1, \dots, w_k, i, w)$
4. imprima w

Luego, $f : \mathbb{C}_c^k \times \mathbb{N} \rightarrow \mathbb{C}_c$ es computable. $\square$

3 Propiedades y ejemplos

Es claro que un número real x es computable si, y sólo si, su parte decimal es computable. Esto nos permite ofrecer las siguientes equivalencias.

Proposición 3.1. *Sea $\alpha \in (0, 1)$. Las siguientes afirmaciones son equivalentes:*

- (a) α es computable.
- (b) La cortadura izquierda $L(\alpha) = \{q \in \mathbb{Q} \mid q < \alpha\}$ es un conjunto recursivo.
- (c) Existe una función computable $f : \mathbb{N} \rightarrow \{0, 1\}$ tal que

$$\alpha = \sum_{n=1}^{\infty} \frac{f(n)}{2^n}.$$

Demostración. Primero veamos que (a) $\Rightarrow$ (b). Notemos que si $\alpha \in \mathbb{Q}$, entonces es inmediato que $L(\alpha)$ es recursivo. Ahora, sea $\alpha \notin \mathbb{Q}$. Dado $q \in \mathbb{Q}$, existe m tal que $|\alpha - q| > \frac{1}{2^m}$. Por tanto, para la aproximación m -ésima q_{m+1} de α se cumple que

$$|q_{m+1} - q| \geq |q_{m+1} - \alpha| - |\alpha - q| > \frac{1}{2^m} - \frac{1}{2^{m+1}} = \frac{1}{2^{m+1}}.$$

Luego, computamos $q_1, q_2, \dots$ hasta hallar q_k que satisfaga $|q_k - q| > \frac{1}{2^k}$. Entonces,

$$|q_k - \alpha| \leq \frac{1}{2^k} < |q_k - q|.$$

Por tanto, $q < \alpha$ sii $q < q_k$. Así, tenemos un procedimiento para decidir si $q < \alpha$; esto es, $L(\alpha)$ es recursivo. Ahora, veamos que $(b) \Rightarrow (c)$. De nuevo, consideremos dos casos. Es un hecho elemental que se pueden computar los bits de cualquier α racional. Para $\alpha \notin \mathbb{Q}$, basta observar que para todo n se cumple que

$$a_{n+1} = 1 \quad \text{sii} \quad \sum_{k=1}^n \frac{a_k}{2^k} + \frac{1}{2^{n+1}} < \alpha.$$

Como $L(\alpha)$ es recursivo, entonces lo anterior describe un procedimiento efectivo para calcular los bits de α . Por último, para $(c) \Rightarrow (a)$ basta tomar $q_n := \sum_{k=1}^n a_k \cdot 2^{-k}$. $\square$

El contenido del enunciado (c) de la proposición 3.1 corresponde a la definición original de número computable que Turing propuso en [8]. Cabe mencionar que las implicaciones $(a) \Rightarrow (b)$ y $(b) \Rightarrow (c)$ no son efectivas, lo cual es de esperarse dado que en las pruebas debemos distinguir entre el caso racional e irracional. Por tanto, las anteriores definiciones no son efectivamente equivalentes. De hecho, se puede mostrar que la implicación $(a) \Rightarrow (c)$ tampoco es efectiva. Esto significa que no existe un procedimiento efectivo que aplicado a un código para un real computable α dé como salida un índice para un algoritmo que genere los *bits* de α (para más detalles ver [1]).

Dado que el conjunto de funciones de aproximación es contable, los conjuntos $\mathbb{C}_c$ y $\mathbb{R}_c$ son contables, sin embargo, es imposible enumerarlos efectivamente. Para probar este notable hecho necesitamos la siguiente noción.

Definición 3.2. Decimos que una sucesión $(z_n)_{n=1}^\infty$ en $\mathbb{C}_c$ es computable si existe una función recursiva $g: \mathbb{N} \rightarrow \mathbb{N}$ tal que $g(n)$ es un código para z_n .

Proposición 3.3. $\mathbb{C}_c$ y $\mathbb{R}_c$ no son efectivamente enumerables.

Demostración. Basta ver que $\mathbb{R}_c$ no es efectivamente enumerable. De hecho, mostremos que si (α_n) es una sucesión computable de números reales computables, entonces existe un número real computable β en $[0, 1]$ tal que $\beta \neq \alpha_n$ para todo n . Construyamos una sucesión encajada de intervalos

$$I_0 := [0, 1] \supset I_1 \supset I_2 \supset \cdots \supset I_n \supset I_{n+1} \supset \cdots$$

tal que para todo $n \geq 1$ se satisface lo siguiente: (i) I_n es el tercio derecho o el tercio izquierdo de I_{n-1} . (ii) $\alpha_k \notin I_n$ para $k = 1, \dots, n$. Suponiendo que hemos construido $I_1, I_2, \dots, I_{n-1}$ con las propiedades requeridas, construimos I_n mediante el siguiente procedimiento: sean J_1 el tercio izquierdo, J_2 el tercio medio y J_3 el tercio derecho de I_{n-1} . Sea x_n el punto medio de I_n . Por hipótesis, podemos computar un número racional $r_n = q_{n+1}^{(n)}$ tal que

$$|\alpha_n - r_n| \leq \frac{1}{2^{n+1}}.$$

Ahora, si $r_n \leq x_n$, hacemos $I_n = J_3$. En caso contrario, hacemos $I_n = J_1$. En cualquier caso, $\alpha_n \notin I_n$. Como la longitud de I_n es $\frac{1}{3^n}$, existe un único número real β tal que

$$\beta \in \bigcap_{n=1}^{\infty} I_n.$$

Por (ii), es inmediato que $\beta \neq \alpha_n$ para todo n . Además la función $f : \mathbb{N} \rightarrow \mathbb{Q}$, dada por $f(n) = b_n = \text{extremo derecho de } I_n$, es claramente computable. Como

$$|\beta - b_n| \leq \frac{1}{3^n} < \frac{1}{2^n}, \quad \text{para todo } n,$$

se sigue que $\beta \in \mathbb{R}_c$. □

De lo anterior se sigue que cualquier intervalo cerrado no degenerado de $\mathbb{R}$ contiene un conjunto no contable de números no computables. Por otro lado, el siguiente lema nos ofrece un criterio muy sencillo para definir números no computables.

Lema 3.4. Sean $A \subseteq \mathbb{N}$ y

$$\alpha := \sum_{n=1}^{\infty} \frac{\chi_A(n)}{3^n} \in [0, 1].$$

Entonces $\alpha \in \mathbb{R}_c$ si y sólo si A es recursivo.

Demostración. Si A es recursivo, entonces $\chi_A : \mathbb{N} \rightarrow \{0, 1\}$ es una función recursiva. Así, para aproximar a α , basta tomar

$$q_n := \sum_{k=1}^n \frac{\chi_A(k)}{3^k}.$$

Ahora, supongamos que $\alpha \in \mathbb{R}_c$ y que (q_n) es una sucesión de racionales que aproxima a 2α . Definamos la función $\psi : \mathbb{N} \rightarrow \{0, 2\}$ por el siguiente procedimiento recursivo: $\psi(0) = 0$ si y sólo si $0 \in A$ y habiendo computado $\psi(0), \dots, \psi(n-1)$, definimos $\psi(n)$ por

$$\psi(n) = \begin{cases} 0 & \text{si } q_{2n+2} < \sum_{k=0}^{n-1} \frac{\psi(k)}{3^k} + \frac{1}{2} \cdot \frac{1}{3^{n-1}}, \\ 2 & \text{si } q_{2n+2} > \sum_{k=0}^{n-1} \frac{\psi(k)}{3^k} + \frac{1}{2} \cdot \frac{1}{3^{n-1}}, \\ \text{indefinido} & \text{en caso contrario.} \end{cases}$$

Claramente, ψ es recursiva parcial. Notemos que para cada n :

$$|2\alpha - q_{2n+2}| \leq \frac{1}{2^{2n+2}} < \frac{1}{2 \cdot 3^n}.$$

Si $\chi_A(n) = 0$, entonces

$$2\alpha \leq \sum_{k=0}^{n-1} \frac{2 \cdot \chi_A(k)}{3^k} + \sum_{k=n+1}^{\infty} \frac{2}{3^k} = \sum_{k=0}^{n-1} \frac{2 \cdot \chi_A(k)}{3^k} + \frac{1}{3^n}.$$

Por tanto,

$$q_{2n+2} < 2\alpha + \frac{1}{2 \cdot 3^n} \leq \sum_{k=0}^{n-1} \frac{2 \cdot \chi_A(k)}{3^k} + \frac{1}{2} \cdot \frac{1}{3^{n-1}}.$$

Similarmente, si $\chi_A(n) = 1$, entonces

$$q_{2n+2} > \sum_{k=0}^{n-1} \frac{2 \cdot \chi_A(k)}{3^k} + \frac{1}{2} \cdot \frac{1}{3^{n-1}}.$$

Es fácil ver, por inducción sobre n , que $\psi(n) = 2 \cdot \chi_A(n)$. $\square$

Es un hecho notable que la equivalencia anterior es efectiva, aunque dicha prueba es más complicada (ver [1]). Ahora, si tomamos $A = \mathbf{K}$, podemos concluir que el correspondiente número real

$$\alpha_{\mathbf{K}} = \sum_{n=1}^{\infty} \frac{\chi_{\mathbf{K}}(n)}{3^n}$$

no es computable. Por otra parte en 1974, apoyándose en la Teoría Algorítmica de la Información, Chaitin definió un número real no computable, de hecho aleatorio, que llamó la *probabilidad de parada* Ω . Debido a sus conexiones con la Incompletez sintáctica, la Aleatoriedad y la insolubilidad algorítmica del Décimo Problema de Hilbert, el número Ω se ha convertido en una de las constantes más célebres de la matemática (para una descripción informal de Ω ver el reciente artículo [3]).

Intuitivamente, Ω representa la “probabilidad” de que cierto tipo de máquina de Turing universal $\mathcal{U}$ (llamada autolimitante) se detenga cuando se le ingresa una entrada u elegida al azar. Es importante observar que Ω depende de la máquina $\mathcal{U}$ (equivalentemente, depende de la gödelización de las máquinas de Turing autolimitantes) que se halla fijado. Así, lo que tenemos realmente es una familia de números $\Omega_{\mathcal{U}}$, no una constante absoluta como π o e . Para un estudio más detallado de Ω y sus relaciones con la lógica y la computabilidad, referimos el lector a [2] o [7].

De otro lado, es natural pensar que las operaciones aritméticas elementales entre números computables sean computables. Sin embargo, este hecho depende de la definición que adoptemos. En otras palabras, podemos mostrar que existe un procedimiento efectivo que aplicado a números computables α, β genera un código para $\alpha + \beta$. Pero tal procedimiento no existe si adoptamos la definición original de Turing (ver [1]). Ahora, dado que las operaciones

de campo entre números complejos están definidas a partir de las operaciones entre números reales, es suficiente restringirnos al caso real (Modificamos ligeramente las pruebas correspondientes que aparecen en [1]).

Teorema 3.5. *Las operaciones de suma, resta y producto de $\mathbb{R}_c \times \mathbb{R}_c$ en $\mathbb{R}_c$ y la operación de división de $\mathbb{R}_c \times (\mathbb{R}_c \setminus \{0\})$ en $\mathbb{R}_c \setminus \{0\}$ son computables.*

Demostración. Veamos que el producto $\cdot : \mathbb{R}_c \times \mathbb{R}_c \rightarrow \mathbb{R}_c$ es computable. Dados $\alpha, \beta \in \mathbb{R}_c$, sean i, j códigos para α y β respectivamente. Luego, para cada k :

$$\begin{aligned} |\alpha \cdot \beta - \varphi_i(k) \cdot \varphi_j(k)| &\leq |\alpha - \varphi_i(k)| \cdot |\beta| + |\beta - \varphi_j(k)| \cdot |\varphi_i(k)| \\ &\leq \frac{1 + |\varphi_j(k)| + |\varphi_i(k)|}{2^k}. \end{aligned}$$

Ahora, consideremos la función parcial recursiva $\Psi : \mathbb{N}^3 \rightarrow \mathbb{N}$ dada por

$$\Psi(i, j, n) = \text{menor } k \in \mathbb{N} \text{ tal que } \frac{1 + |\varphi_j(k)| + |\varphi_i(k)|}{2^k} \leq \frac{1}{2^n}.$$

Como

$$1 + |\varphi_i(k)| + |\varphi_j(k)| \leq 1 + (|\alpha| + \frac{1}{2^k}) + (|\beta| + \frac{1}{2^k}) \leq |\alpha| + |\beta| + 3,$$

es claro que

$$\lim_{k \rightarrow \infty} \frac{1 + |\varphi_j(k)| + |\varphi_i(k)|}{2^k} = 0.$$

Luego, Ψ es total recursiva. Luego, la función $f : \mathbb{N}^3 \rightarrow \mathbb{Q}$, dada por

$$f(i, j, n) = \varphi_i(\Psi(i, j, n)) \cdot \varphi_j(\Psi(i, j, n)),$$

también es total recursiva. Así, por el teorema s - m - n , existe una función total recursiva $\theta : \mathbb{N}^2 \rightarrow \mathbb{N}$ tal que

$$\varphi_{\theta(i,j)}(n) = f(i, j, n), \quad \text{para todo } n.$$

Basta ver que $\theta(i, j)$ es un código para $\alpha \cdot \beta$. En efecto,

$$\begin{aligned} |\alpha \cdot \beta - \varphi_{\theta(i,j)}(n)| &= |\alpha \cdot \beta - \varphi_i(\Psi(i, j, n)) \cdot \varphi_j(\Psi(i, j, n))| \\ &\leq \frac{1 + |\varphi_j(\Psi(i, j, n))| + |\varphi_i(\Psi(i, j, n))|}{2^{\Psi(i,j,n)}} \\ &\leq \frac{1}{2^n}. \end{aligned}$$

Luego, el producto en $\mathbb{R}_c$ es una operación computable. De forma análoga, se puede mostrar que la suma y la diferencia en $\mathbb{R}_c$ son computables. En cuanto a la división en $\mathbb{R}_c$, dada la computabilidad del producto y la proposición 2.7 (c),

es suficiente probar que la operación $\cdot^{-1} : \mathbb{R}_c \setminus \{0\} \rightarrow \mathbb{R}_c \setminus \{0\}$ es computable. Sean α un número real computable no nulo y e un correspondiente código para α . Teniendo presente la enumeración (φ_m) de todas las funciones de $\mathbb{N}$ en $\mathbb{Q}$, podemos considerar la función parcial recursiva $\psi : \mathbb{N} \rightarrow \mathbb{N}$ dada por

$$\psi(m) = \text{menor } k \geq 2 \text{ tal que } \frac{1}{2^{k-2}} \leq |\varphi_m(k)|.$$

Ahora, definamos una función parcial recursiva $\Psi : \mathbb{N}^2 \rightarrow \mathbb{N}$ por

$$\Psi(m, n) = \begin{cases} \frac{1}{\varphi_m(2\psi(m) + n - 2)} & \text{si } \varphi_m(2\psi(m) + n - 2) \\ & \text{está definido y es no nulo,} \\ \text{indefinido} & \text{en caso contrario.} \end{cases}$$

Aplicando el teorema *s-m-n*, existe una función total recursiva $\theta : \mathbb{N} \rightarrow \mathbb{N}$ tal que $\varphi_{\theta(m)} = \Psi(m, \cdot)$ para cada m . Como $\alpha \neq 0$, existe k_0 tal que $\frac{5}{2^{k_0}} \leq |\alpha|$. Luego,

$$|\varphi_e(k_0)| \geq |\alpha| - \frac{1}{2^{k_0}} \geq \frac{5}{2^{k_0}} - \frac{1}{2^{k_0}} = \frac{1}{2^{k_0-2}}$$

y así, $\psi(e)$ está definido. Además, para todo $k \geq \psi(e)$ tenemos que

$$\begin{aligned} |\varphi_e(k)| &\geq |\varphi_e(\psi(e))| - |\alpha - \varphi_e(\psi(e))| - |\alpha - \varphi_e(k)| \\ &\geq \frac{1}{2^{\psi(e)-2}} - \frac{1}{2^{\psi(e)}} - \frac{1}{2^{\psi(e)}} = \frac{1}{2^{\psi(e)-1}}. \end{aligned}$$

Por tanto,

$$\frac{1}{2^{\psi(e)-1}} \leq \lim_{k \rightarrow \infty} |\varphi_e(k)| = |\alpha| \quad \text{y} \quad \frac{1}{|\alpha|} \leq 2^{\psi(e)-1}.$$

Dado que $\psi(e) \geq 2$, para cada $n \in \mathbb{N}$ se tiene que $2\psi(e) + n - 2 \geq \psi(e)$ y, por tanto,

$$0 < \frac{1}{2^{\psi(e)-1}} \leq |\varphi_e(2\psi(e) + n - 2)|.$$

Luego, $\varphi_{\theta(e)}(n) = \Psi(e, n)$ está definido y cumple que

$$\left| \frac{1}{\alpha} - \varphi_{\theta(e)}(n) \right| = \frac{|\alpha - \varphi_e(2\psi(e) + n - 2)|}{|\alpha| \cdot |\varphi_e(2\psi(e) + n - 2)|} \leq \frac{2^{\psi(e)-1} \cdot 2^{\psi(e)-1}}{2^{2\psi(e)+n-2}} = \frac{1}{2^n}.$$

Esto muestra que $\theta(e)$ es un código para $\frac{1}{\alpha}$. □

Corolario 3.6. *Las operaciones de suma, resta y producto de $\mathbb{C}_c \times \mathbb{C}_c$ en $\mathbb{C}$ y la operación de división de $\mathbb{C}_c \times (\mathbb{C}_c \setminus \{0\})$ en $\mathbb{C} \setminus \{0\}$ son computables.*

4 Orden y algoritmos de decisión en $\mathbb{R}_c$

Notemos que, por el teorema 3.5, $\mathbb{R}_c$ es un subcampo ordenado de $\mathbb{R}$. Sin embargo, en general es computacionalmente imposible comparar dos reales computables arbitrarios. La principal dificultad radica en el hecho en que un número real computable sólo se puede describir de forma indirecta a través de un código (o una función de aproximación). Intuitivamente, para comparar dos números reales computables uno necesitaría conocer de antemano toda la expansión binaria de los números en cuestión. Lo cual nos lleva a la insolubilidad algorítmica del problema de la parada, como lo ilustra el siguiente resultado fundamental.

Lema 4.1. *El problema de decidir si una función cualquiera $\varphi_i : \mathbb{N} \rightarrow \{0, 1\}$ recursiva total es idénticamente cero es algorítmicamente insoluble.*

Demostración. Supongamos que existe una función recursiva $\theta : \mathbb{N} \rightarrow \{0, 1\}$ tal que si $\varphi_i : \mathbb{N} \rightarrow \{0, 1\}$ es total recursiva, entonces $i \in \text{dom}(\theta)$ y

$$\theta(i) = 0 \quad \text{sii} \quad \varphi_i \equiv 0.$$

Definamos una función total recursiva $H : \mathbb{N}^2 \rightarrow \{0, 1\}$ por

$$H(m, n) = \begin{cases} 1, & \text{si } M_m(m) \text{ para a lo sumo en } n \text{ pasos} \\ 0, & \text{en caso contrario.} \end{cases}$$

Por el teorema s - m - n existe una función total recursiva $s : \mathbb{N} \rightarrow \mathbb{N}$ tal que $\varphi_{s(m)} = H(m, \cdot)$. Como $\varphi_{s(m)} : \mathbb{N} \rightarrow \{0, 1\}$ es recursiva, entonces $s(m) \in \text{dom}(\theta)$ y

$$\theta(s(m)) = 0 \quad \text{sii} \quad \varphi_{s(m)} \equiv 0 \quad \text{sii} \quad M_m(m) \text{ nunca para.}$$

Si tomamos $\psi = \theta \circ s : \mathbb{N} \rightarrow \{0, 1\}$, tenemos que ψ es recursiva y que

$$\psi(m) = 0 \quad \text{sii} \quad m \in \mathbf{K}.$$

Por tanto, $\mathbf{K}$ sería recursivo; absurdo. □

Proposición 4.2. *No existe un algoritmo que, aplicado a cualesquiera $\alpha, \beta \in \mathbb{R}_c$, decida si $\alpha > \beta$.*

Demostración. Notemos que es suficiente mostrar que no existe un algoritmo que, aplicado a un número real computable, decida si es positivo o negativo. Luego, supongamos que existe una función parcial recursiva $\theta : \mathbb{N} \rightarrow \{0, 1\}$ tal que para todo $\alpha \in \mathbb{R}_c$ si e es un código para α , entonces $e \in \text{dom}(\theta)$ y

$$\theta(e) = 1 \quad \text{sii} \quad \alpha > 0.$$

Ahora, dado i tal que $\varphi_i : \mathbb{N} \rightarrow \{0, 1\}$ es una función total recursiva, definamos

$$\alpha = \sum_{n=0}^{\infty} \frac{\varphi_i(n)}{2^n}.$$

Entonces $\alpha \in \mathbb{R}_c$ y la función $f : \mathbb{N} \rightarrow \mathbb{Q}$ dada por

$$f(k) = \sum_{n=0}^k \frac{\varphi_i(n)}{2^n} \quad (k \in \mathbb{N}).$$

es una función de aproximación para α . Elijamos un índice m tal que $\nu^{-1} \circ f = \varphi_m$. Así, m es un código para α y $m \in \text{dom}(\theta)$. Si $\theta(m) = 1$, entonces $\alpha > 0$ y, por tanto, existe n tal que $\varphi_i(n) = 1$. Si $\theta(m) = 0$, entonces $\alpha \leq 0$. Luego, $\varphi_i(n) = 0$, para todo n . Es claro que el anterior razonamiento es independiente del índice m elegido para la función de aproximación f para α y que

$$\psi(i) = \theta(m)$$

define una función parcial recursiva $\psi : \mathbb{N} \rightarrow \{0, 1\}$ tal que

$$\psi(i) = 0 \quad \text{sii} \quad \varphi_i \equiv 0.$$

Absurdo, pues se contradice el lema 4.1. □

De acuerdo a la prueba de la proposición 4.2 no existe una máquina de Turing que aplicada a la entrada e (el código para una función que aproxima a α) dé como salida **Sí** cuando $\alpha > 0$ o **No** si $\alpha \leq 0$. Intuitivamente, el problema estriba en que la máquina codificada por e podría generar sólo ceros como aproximación para α y no es claro cuánto tiempo debemos esperar para decidir que la máquina *nunca* generará una aproximación que obligue a que α sea positivo.

Proposición 4.3. *No existe un algoritmo que, aplicado a cualesquiera $\alpha, \beta \in \mathbb{R}_c$, decida si $\alpha = \beta$.*

Demostración. Basta mostrar que no existe una función parcial recursiva $\theta : \mathbb{N} \rightarrow \{0, 1\}$ tal que para todo $\alpha \in \mathbb{R}_c$ si e es un código para α , entonces $e \in \text{dom}(\theta)$ y $\theta(e) = 0$ sii $\alpha = 0$. En caso contrario, dado i tal que $\varphi_i : \mathbb{N} \rightarrow \{0, 1\}$ es una función total recursiva, consideramos el número real computable

$$\alpha = \sum_{n=0}^{\infty} \frac{\varphi_i(n)}{2^n}$$

con código m . Si $\theta(m) = 0$, entonces $\alpha = 0$. Luego, $\varphi_i(n) = 0$, para todo n . Si $\theta(m) = 1$, entonces $\alpha \neq 0$ y, por tanto, existe n tal que $\varphi_i(n) = 1$. Así, $\psi(i) = \theta(m)$ define una función parcial recursiva $\psi : \mathbb{N} \rightarrow \{0, 1\}$ tal que

$$\psi(i) = 0 \quad \text{sii} \quad \varphi_i \equiv 0.$$

Absurdo, de nuevo se contradice el lema 4.1. $\square$

En contraste con la insolubilidad algorítmica de los problemas previos, es posible mostrar que existe un procedimiento efectivo que, aplicado a dos números reales computables distintos, decide cuál es mayor.

Teorema 4.4. *Existe un algoritmo que, aplicado a $\alpha, \beta \in \mathbb{R}_c$ tales que $\alpha \neq \beta$, decide si $\alpha < \beta$.*

Demostración. Dado que $\alpha < \beta$ sii $\alpha - \beta < 0$ y que la diferencia en $\mathbb{R}_c$ es computable (teorema 3.5), es suficiente mostrar que existe un algoritmo que, aplicado a un número real computable no nulo, decide si es positivo o negativo. Ahora, dado un código e de un número arbitrario α en $\mathbb{R}_c \setminus \{0\}$, aplicamos el siguiente procedimiento:

1. Genere el menor n tal que $|\varphi_e(n+1)| > \frac{1}{2^{n+1}}$.
2. Si $\varphi_e(n+1) > 0$, entonces
 imprima el número es positivo y pare;
 en caso contrario
 imprima el número es negativo y pare.

Como $\alpha \neq 0$, existe n_0 tal que $|\alpha| > 2^{-n_0}$. Además $|\alpha - \varphi_e(n_0+1)| \leq 2^{-n_0-1}$, implica que

$$|\varphi_e(n_0+1)| \geq |\alpha| - \frac{1}{2^{n_0+1}} > \frac{1}{2^{n_0}} - \frac{1}{2^{n_0+1}} = \frac{1}{2^{n_0+1}}.$$

Por tanto, el paso 1 es efectivo. Ahora, notemos que

$$-\frac{1}{2^{n+1}} \leq \alpha - \varphi_e(n+1) \leq \frac{1}{2^{n+1}}.$$

Si $\varphi_e(n+1) > 0$, entonces

$$\alpha \geq \varphi_e(n+1) - \frac{1}{2^{n+1}} > \frac{1}{2^{n+1}} - \frac{1}{2^{n+1}} = 0.$$

Similarmente, si $\varphi_e(n+1) < 0$, entonces $\alpha < 0$. Por tanto, el anterior procedimiento decide efectivamente si α es positivo ó negativo. $\square$

Por último, mencionamos que no existe un procedimiento efectivo que aplicado a un código para un racional q decida si $q = 0$. Similarmente, tampoco existe un algoritmo que aplicado a un $\alpha \in \mathbb{R}_c$ determine si α es racional ó irracional (ver [1]).

5 Aplicaciones al análisis

Uno de los teoremas clásicos del análisis real afirma que toda sucesión monótona y acotada es convergente. Como consecuencia de la insolubilidad algorítmica del problema de la parada, este principio fundamental es falso en $\mathbb{R}_c$. Para justificar este hecho, introducimos la siguiente definición y unos resultados preliminares.

Definición 5.1. Una sucesión (z_n) en $\mathbb{C}_c$ converge efectivamente a $z \in \mathbb{C}$ si existe una función total recursiva $h : \mathbb{N} \rightarrow \mathbb{N}$ tal que para todo m :

$$|z - z_n| \leq 2^{-m}, \quad \text{si } n \geq h(m).$$

Proposición 5.2. Si (z_n) es una sucesión computable en $\mathbb{C}_c$ que converge efectivamente a z , entonces $z \in \mathbb{C}_c$.

Demostración. Por hipótesis, existen funciones totales recursivas $g, h : \mathbb{N} \rightarrow \mathbb{N}$ tales que para cada $n, m : \varphi_{g(n)} : \mathbb{N} \rightarrow \mathbb{Q}$ es una función de aproximación para z_n y si $n \geq h(m)$, entonces

$$|z - z_n| \leq 2^{-m}.$$

Ahora, definamos una función total recursiva $f : \mathbb{N} \rightarrow \mathbb{Q}$ por

$$f(n) \equiv \varphi_{g \circ h(n+1)}(n+1).$$

Luego, para cada n ,

$$|z - f(n)| \leq |z - z_{h(n+1)}| + |z_{h(n+1)} - \varphi_{g(h(n+1))}(n+1)| \leq \frac{1}{2^n}.$$

Por tanto, $z \in \mathbb{C}_c$. □

La siguiente proposición es un caso particular de un teorema de Calude-Hertling. Para la versión general, ver [2].

Proposición 5.3. Toda sucesión computable monótona de números racionales que converge a un número real computable converge efectivamente.

Demostración. Sea (a_n) una sucesión computable monótona de racionales que converge a $\alpha \in \mathbb{R}_c$. Sea (q_n) una sucesión de racionales que aproximen a α . Dado $m \in \mathbb{N}$ existe k tal que

$$|\alpha - a_k| \leq \frac{1}{2^{m+2}}.$$

Para este k se cumple que

$$|a_k - q_{m+2}| \leq |a_k - \alpha| + |\alpha - q_{m+2}| \leq \frac{1}{2^{m+1}}.$$

Luego, podemos definir una función total recursiva $h : \mathbb{N} \rightarrow \mathbb{N}$ por

$$h(m) = \text{menor } k \in \mathbb{N} \text{ tal que } |a_k - q_{m+2}| \leq \frac{1}{2^{m+1}}.$$

Por la monotonía de (a_n) , para todo m y todo $n \geq h(m)$ se tiene que

$$|\alpha - a_n| \leq |\alpha - a_{h(m)}| \leq |\alpha - q_{m+2}| + |q_{m+2} - a_{h(m)}| < \frac{1}{2^m}.$$

Por tanto, (a_n) converge efectivamente a α . $\square$

Teorema 5.4 (Specker). *Existe una sucesión computable $(a_n) \subseteq \mathbb{Q}$ estrictamente creciente en $[0, 1]$ que no converge efectivamente.*

Demostración. Sea $f : \mathbb{N} \rightarrow \mathbf{K}$ una enumeración efectiva de $\mathbf{K}$ y definamos

$$a_n \equiv \sum_{k=0}^n \frac{1}{2^{f(k)+1}} \in \mathbb{Q}.$$

Claramente, (a_n) es una sucesión computable estrictamente creciente en $[0, 1]$ que converge a un número real α . Supongamos que (a_n) converge efectivamente. Luego, sea $h : \mathbb{N} \rightarrow \mathbb{N}$ una función total recursiva tal que

$$|\alpha - a_n| \leq \frac{1}{2^m}, \quad \text{si } n \geq h(m).$$

Notemos que si $n > h(m+3)$, entonces

$$\frac{1}{2^{f(n)+1}} \leq a_n - a_{h(m+3)} \leq |a_n - \alpha| + |\alpha - a_{h(m+3)}| < \frac{1}{2^{m+1}}.$$

Por tanto, $m < f(n)$. Así, se sigue que $m \in \mathbf{K}$ sii existe $n \leq h(m+3)$ tal que $f(n) = m$. Comparando m con

$$f(1), f(2), \dots, f(h(m+3)),$$

podemos decidir si $m \in \mathbf{K}$. Pero esto es imposible, dado que $\mathbf{K}$ no es recursivo. $\square$

Un número real para el cual exista una sucesión monótona computable de números racionales que converge a él, pero no efectivamente, se denomina *número de Specker*. Por la proposición 5.3 se sigue que los números de Specker son no computables. También podemos concluir que existen sucesiones monótonas y acotadas en $\mathbb{R}_c$ que convergen a números no computables.

Definición 5.5. *Para cada $n \in \mathbb{N}$, sea $f_n : \mathbb{C}_c \rightarrow \mathbb{C}_c$ una función parcial computable. Decimos que la sucesión $(f_n)_{n=0}^\infty$ es computable si la función parcial $(n, z) \mapsto f_n(z)$ de $\mathbb{N} \times \mathbb{C}_c$ en $\mathbb{C}_c$ es computable.*

Podemos generalizar la proposición 5.2 y probar que toda función analítica cuya representación en series de potencias tenga coeficientes computables restringida a cualquier disco cerrado de su dominio es una función computable.

Teorema 5.6. *Sean $f, f_0, f_1, \dots$ funciones parciales complejas con dominio $D \subseteq \mathbb{C}$. Supongamos que existe una función total recursiva $h : \mathbb{N} \rightarrow \mathbb{N}$ tal que*

$$|f(z) - f_n(z)| \leq \frac{1}{2^m}, \quad \text{si } z \in D \text{ y } n \geq h(m).$$

Si (f_n) es una sucesión computable de funciones computables, entonces f es una función parcial computable de $\mathbb{C}_c$ en $\mathbb{C}_c$.

Demostración. Por hipótesis, existe una función parcial recursiva $\theta : \mathbb{N}^2 \rightarrow \mathbb{N}$ tal que si $\varphi_e : \mathbb{N} \rightarrow \mathbb{Q}^2$ es una función de aproximación para $z \in D$, entonces $(n, e) \in \text{dom}(\theta)$ y $\varphi_{\theta(n,e)} : \mathbb{N} \rightarrow \mathbb{Q}^2$ es una función de aproximación para $f_n(z)$. Definamos una función parcial recursiva $\Psi : \mathbb{N}^2 \rightarrow \mathbb{Q}^2$ por

$$\Psi(e, n) \equiv \varphi_{\theta(h(n+1), e)}(n+1).$$

Aplicando el teorema *s-m-n*, existe una función total recursiva $s : \mathbb{N} \rightarrow \mathbb{N}$ tal que $\varphi_{s(e)} = \Psi(e, \cdot)$ para cada e . Ahora, sean $z \in D$ computable y e un código para z . Entonces, para cada n :

$$\begin{aligned} |f(z) - \varphi_{s(e)}(n)| &\leq |f(z) - f_{h(n+1)}(z)| + \\ &|f_{h(n+1)}(z) - \varphi_{\theta(h(n+1), e)}(n+1)| \leq \frac{1}{2^{n+1}} + \frac{1}{2^{n+1}} = \frac{1}{2^n}. \end{aligned}$$

Por tanto, $s(e)$ es un código para $f(z)$. Así, f es una función parcial computable de $D \cap \mathbb{C}_c$ en $\mathbb{C}_c$. $\square$

Teorema 5.7. *Sea*

$$f(z) = \sum_{n=0}^{\infty} \alpha_n \cdot z^n$$

una serie de potencias con disco abierto de convergencia Ω . Si $(\alpha_n) \subseteq \mathbb{C}_c$ es una sucesión computable, entonces f restringida a cualquier disco cerrado $\Delta \subseteq \Omega$ es una función computable.

Demostración. Dado un disco cerrado Δ , tomamos $z_0 \in \Omega$ tal que $\Delta \subseteq D(0, r)$, donde $r = |z_0|$. Luego, existen racionales $b \in \Omega$ y $M > 0$ tales que $r < b$ y

$$\sum_{n=0}^{\infty} |\alpha_n| \cdot b^n \leq M.$$

Ahora, escogemos n_0 tal que :

$$\left(\frac{r}{b}\right)^{n_0} \leq \frac{1}{2}$$

y definimos

$$f_n(z) = \sum_{k=0}^{n+n_0} \alpha_k \cdot z^k, \quad (n \in \mathbb{N}).$$

Sea ahora h la función computable definida como sigue: $h(m)$ es el mínimo entero $\geq n_0$ tal que $2^{h(m)} \geq M \cdot 2^m$. Entonces, para todo $z \in D(0, r)$ y todo $n \geq h(m)$, se cumple que

$$\begin{aligned} |f(z) - f_n(z)| &\leq \sum_{k=n+n_0+1}^{\infty} |\alpha_k| \cdot |z|^k = \sum_{k=n+n_0+1}^{\infty} (|\alpha_k| \cdot b^k) \cdot \left(\frac{|z|}{b}\right)^k \\ &\leq M \cdot \sum_{k=n_0+n+1}^{\infty} \left(\frac{r}{b}\right)^k = M \cdot \sum_{k=n+1}^{\infty} \left[\left(\frac{r}{b}\right)^{n_0}\right]^k \\ &\leq M \cdot \sum_{k=n+1}^{\infty} \left(\frac{1}{2}\right)^k = \frac{M}{2^n} \leq \frac{1}{2^m}. \end{aligned}$$

Aplicando el teorema 2.9 y el corolario 3.6 (b), es claro que (f_n) es una sucesión computable de funciones (totales) computables. Luego, se satisfacen las condiciones del teorema anterior con $D = D(0, r)$. Por tanto, f restringida a Δ es computable. $\square$

Corolario 5.8. *Sea*

$$f(z) = \sum_{n=0}^{\infty} \alpha_n \cdot z^n, \quad \alpha_n \in \mathbb{C}_c,$$

una serie de potencias con disco abierto de convergencia Ω . Si (α_n) es una sucesión computable, entonces para todo $z \in \Omega$ computable se cumple que $f(z)$ es computable.

Como otra consecuencia directa del teorema 5.7 podemos concluir que las funciones elementales trascendentales (e^z , $\cos z$, $\arcsin z$, $\log z$, etc) aplicadas a números computables generan números computables. De este modo, las constantes *trascendentes* usuales que aparecen en el análisis, como e , π , $\log 2$, entre otras, son computables.

6 Aplicaciones al álgebra

En esta sección, nos proponemos mostrar que el campo $\mathbb{C}_c$ es algebraicamente cerrado y contiene a los números algebraicos. Iniciamos probando la siguiente versión efectiva del teorema del valor intermedio para polinomios en $\mathbb{R}_c$.

Proposición 6.1. *Sean $p(x)$ un polinomio en $\mathbb{R}_c[x]$ y $\alpha < \beta$ en $\mathbb{R}_c$ tales que $p(\alpha) \cdot p(\beta) < 0$. Si $p(x)$ es monótono en el intervalo $[\alpha, \beta]$, entonces existe un único $\gamma \in \mathbb{R}_c$ tal que $p(\gamma) = 0$.*

Demostración. Sin pérdida de generalidad, supongamos que $p(x)$ es creciente en $[\alpha, \beta]$. Por el teorema del valor intermedio, existe un único $\gamma \in \mathbb{R}$ tal que $p(\gamma) = 0$. Veamos que $\gamma \in \mathbb{R}_c$. Consideremos el siguiente procedimiento de bisección. Dado el intervalo $[\alpha, \beta]$ calculamos su punto medio $\alpha_1 := \frac{1}{2}(\alpha + \beta)$. Ahora, tomamos los subintervalos resultantes $[\alpha, \alpha_1]$, $[\alpha_1, \beta]$ y sus correspondientes puntos medios, generando así cuatro subintervalos. Si iteramos este procedimiento, generamos una familia *infinita* de subintervalos de $[\alpha, \beta]$. Sea D el conjunto formado por los extremos de dichos subintervalos. Es claro que $D \subseteq \mathbb{R}_c$. Se presentan dos casos: Caso 1. $\gamma \in D$. Obviamente, $\gamma \in \mathbb{R}_c$. Caso 2. $\gamma \notin D$. Por tanto, $p(d) \neq 0$, para todo $d \in D$. Entonces, definiremos una sucesión computable (α_n) de aproximaciones de acuerdo al siguiente procedimiento.

1. Haga $L := \alpha$ y $R := \beta$.
2. Para k de 1 hasta n
 haga $\alpha_k := \frac{L+R}{2}$ y compute $p(\alpha_k)$
 si $p(\alpha_k) < 0^1$, entonces
 haga $L := \alpha_k$
 en caso contrario
 haga $R := \alpha_k$
3. Genere α_n .

Es claro que $\alpha_n \in D$ y que

$$|\alpha_n - \gamma| \leq \frac{|\alpha - \beta|}{2^n}.$$

Por tanto, $(\alpha_n)_{n=1}^\infty$ es una sucesión de números computables que converge efectivamente a γ . Por la proposición 5.2, γ es computable. $\square$

Teorema 6.2. $\mathbb{A} \subseteq \mathbb{C}_c$. *Esto es, todo número algebraico es computable.*

Demostración. Por la definición de número algebraico, se tiene que $z \in \mathbb{A}$ si y sólo si $\bar{z} \in \mathbb{A}$. Dado que $\mathbb{A}$ es un campo, se cumple entonces que

$$z \in \mathbb{A} \quad \text{si y sólo si} \quad \operatorname{Re} z, \operatorname{Im} z \in \mathbb{A}.$$

Por tanto, es suficiente probar que todo real algebraico α en $[0, 1]$ es computable. Mostremos por inducción sobre el grado de cualquier polinomio $p(x) \in \mathbb{Z}[x]$ que todas sus raíces en $[0, 1]$ son computables. Cuando $n = 1$, $p(x) = ax + b$, donde $a, b \in \mathbb{Z}$. En este caso, $\alpha = -\frac{b}{a} \in \mathbb{Q}$ es obviamente computable. Supongamos que para todo polinomio con grado menor que n se cumple el enunciado y sea $p(x)$ un polinomio de grado n . Como $p'(x)$, la *derivada* de p , es un polinomio de grado $n - 1$, por hipótesis inductiva,

sus raíces en $[0, 1]$ son reales computables que se pueden ordenar en forma creciente:

$$\alpha_0 = 0 \leq \alpha_1 < \alpha_2 < \cdots < \alpha_{r-1} \leq 1 = \alpha_r.$$

Luego, si $\alpha \in [0, 1]$ es una raíz de $p(x)$, es claro que existe k tal que $\alpha \in [\alpha_k, \alpha_{k+1}]$. Si α es uno de los extremos, entonces α es computable. En caso contrario, α es el único cero de $p(x)$ en $[\alpha_k, \alpha_{k+1}]$, dado que $p(x)$ es monótono en $[\alpha_k, \alpha_{k+1}]$. Además $\alpha_k, \alpha_{k+1} \in \mathbb{R}_c$ y $p(\alpha_k) \cdot p(\alpha_{k+1}) < 0$. Por la proposición 6.1, α es computable. $\square$

Proposición 6.3. $\mathbb{R}_c$ es un campo real cerrado.

Demostración. Primero mostremos que si $\alpha \in \mathbb{R}_c^+$, entonces α tiene una raíz cuadrada en $\mathbb{R}_c$. Para ello, basta aplicar la proposición 6.1 al polinomio $p(x) = x^2 - \alpha \in \mathbb{R}_c[x]$ en el intervalo $[0, \alpha + 1]$. Ahora, consideremos un polinomio $p(x) \in \mathbb{R}_c[x]$ de grado impar y veamos que p tiene un cero en $\mathbb{R}_c$. Como el grado de $p(x)$ es impar, p toma valores positivos y negativos (arbitrariamente grandes). Por el teorema del valor intermedio, p tiene al menos una raíz real. Podemos suponer, sin pérdida de generalidad, que $\lim_{x \rightarrow -\infty} p(x) = -\infty$. Sea γ el primer cero de $p(x)$ en cierto intervalo $[\alpha, \beta]$ tal que

$$p(\alpha) < 0 \quad \text{y} \quad p(\beta) > 0.$$

Por continuidad, podemos suponer que $\alpha, \beta \in \mathbb{Q}$ y que γ es el único cero de p en $[\alpha, \beta]$. Como $p(x)$ es creciente en $[\alpha, \beta]$, por la proposición 6.1, $\gamma \in \mathbb{R}_c$. $\square$

De la anterior proposición, podemos concluir que la extensión $\mathbb{R}_c[i]$ es un campo algebraicamente cerrado (ver [6]). Además, es claro que

$$\mathbb{R}_c[i] = \{a + bi \mid a, b \in \mathbb{R}_c\} = \mathbb{C}_c.$$

Así, hemos obtenido el siguiente

Teorema 6.4. $\mathbb{C}_c$ es un campo algebraicamente cerrado.

Agradecimientos. Este trabajo hace parte del proyecto **Aleatoriedad y Lógica** (código QUIPÚ: 030802734) financiado por la Dirección de Investigación, sede Medellín (DIME), de la Universidad Nacional de Colombia.

Referencias

- [1] Douglas S. Bridges. Computability. A Mathematical Sketchbook. Springer-Verlag, New York, 1994.
- [2] Cristian S. Calude. Information and Randomness - An Algorithmic Perspective, Springer-Verlag, Berlin, 2002.

- [3] Gregory Chaitin. The Limits of Reason. Scientific American, 294, No. 3 (March 2006), p. 74-81.
- [4] Nigel J. Cutland. Computability. Cambridge University Press, Cambridge, 1997.
- [5] Martin Davis (ed). The Undecidable, Raven Press, New York, 1965.
- [6] Serge Lang. Algebra. Springer-Verlag, New York, 2005.
- [7] Johany A. Suárez. El Número Omega. Información, Incompletez y Aleatoriedad. Tesis de Grado. Universidad de Antioquia, Medellín. 2003.
- [8] Alan M. Turing. On Computable Numbers, with an Application to the Entscheidungsproblem. Davis, M (ed) The Undecidable. p 115-154, 1965.

Dirección de los autores: Carlos M. Parra , Universidad Nacional de Colombia, Sede Medellín, cmparra@unalmed.edu.co — Johany A. Suárez , Universidad Nacional de Colombia, Sede Medellín, jasuaezr@unalmed.edu.co