Course Notes for MPRI Course 2.33.1: Recursive Analysis

Olivier Bournez

Version of November 2, 2018.

Chapter 1

Preliminaries

These are Course Notes for MPRI Course 2.33.1.
Theories of Computation.

There are highly based on:

- [1] and [3]
- [9]
- [4]

Any comment (even about orthography) welcome: send an email to bournez@lix.polytechnique.fr

Chapter 2

The notion of Computable Real

2.1 Historical origins

The following sentence can be found in the paper from Turing in 1936 introducing Turing machines [8]:

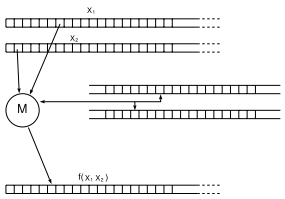
> "The "computable" numbers may be described briefly as the real numbers whose expressions as a decimal are calculable by a finite means."

Hence, it can be stated that the ideas of recursive analysis's approach are as old as computability theory and Turing machines.

2.2 Classical Computability

2.2.1 Turing machines

We assume the reader familiar with Turing machines.



The elementary operations a machine can perform are:

1. move some head on working tape one position to the left or right;
2. move some head on input tape or output tape right;

3. write a symbol $a \in \Sigma$ on the position under the head;

4. compare the symbol under some head with the symbol $a \in \Sigma$.

In other words, we assume the output tape to be **write-only** (the head cannot move to the left).

2.2.2 Basic notions from Computability Theory

- A function $f : \mathbb{N} \to \mathbb{N}$, or $f : \mathbb{N} \to \mathbb{Q}$, or $f : \mathbb{Q} \to \mathbb{N}$, or $f : \mathbb{Q} \to \mathbb{Q}$ is said *computable*, if there exists a Turing machine which can transfer each number $n \in \mathbb{N}$, encoded in the input tape, into the corresponding function value $f(n)$, which is to be encoded on the output tape, in finite time.

- A set $A \subset \mathbb{N}$ or $A \subset \mathbb{Q}$ is said *computable* or *recursive*, if its characteristic function $\chi_A : \mathbb{N} \to \mathbb{N}$ or $\chi_A : \mathbb{Q} \to \mathbb{N}$ is computable.

- A set $A \subset \mathbb{N}$ or $A \subset \mathbb{Q}$ is said *computably enumerable*, or *recursively enumerable* if it is empty or if there exists a computable function $f : \mathbb{N} \to \mathbb{N}$ or $f : \mathbb{N} \to \mathbb{Q}$ respectively such that $A = range(f)$.

2.3 Computable Real Numbers

2.3.1 Definition

Definition 1 *A real number is called* computable *if there exists a Turing machine wich can produce its decimal expansion (without input).*

2.3.2 Characterization

Theorem 1 *Let $x \in \mathbb{R}$. The following are equivalent:*

1. *x is a computable real number.*

2. *there exists a Turing machine wich can produce its binary expansion (without input).*

3. *there exists a computable sequence of rational numbers $(q_n)_{n\in\mathbb{N}}$ which converges rapidly to x, i.e $|q_i - x| < 2^{-i}$.*

4. *There exists a computable sequence of rational shrinking intervals enclosing only x: that is to say, there exists two computable sequences $(a_n)_{n\in\mathbb{N}}$ and $(b_n)_{n\in\mathbb{N}}$of rational numbers with*

$$a_0 < \cdots < a_n < a_{n+1} < \ldots x \cdots < b_{n+1} < b_n < \cdots < b_0$$

   with $\lim_{n\to\infty} a_n = \lim_{n\to\infty} b_n$.

5. *$\{q \in \mathbb{Q} : q < x\}$ is a recursive (= decidable) set.*

6. *there exists a recursive set* $A \subset \mathbb{N}$ *and an integer* z *such that* $x = z + \sum_{i \in A} 2^{-i-1}$.
7. x *is rational (then it has a finite continued fraction expansion) or* x *admits a computable continued fraction expansion, i.e. there is some integer* z *and some computable* $f : \mathbb{N} \to \mathbb{N}$ *such that*

$$x = z + \cfrac{1}{f(0) + \cfrac{1}{f(1) + \cfrac{1}{f(2) + \cfrac{1}{f(3) + \cdots}}}}.$$

2.3.3 Proof that 5. $\Leftrightarrow$ 3.

- We sketch the proof of $\{q \in \mathbb{Q} : q < x\}$ is a recursive set iff $\exists \ \ (q_n)_{n \in \mathbb{N}}$ with $|q_i - x| < 2^{-i}$.
  - $\Rightarrow$. For each i determine some $q_i \in \mathbb{Q}$ with $q_i < x$ and not $q_i + 2^{-i-1} < x$. These properties are decidable and a suitable i always exists. It follows that $q_i < x \leq q_i + 2^{-i-1}$ and thus $|q_i - x| < 2^{-i}$.
  - $\Leftarrow$.
    1. Case $x \in \mathbb{Q}$. It is easy to decide $q < x$ for a given $q \in \mathbb{Q}$.
    2. Case $x \notin \mathbb{Q}$. For given $q \in \mathbb{Q}$, we determine some $i \in \mathbb{N}$ such that
    $$|q_i - x| < 2^{-i} \text{ and } |q_i - q| > 2^{-i}.$$
    Such an i exists because $x \notin \mathbb{Q}$. Then $q < x$ iff $q < q_i$, which is decidable.
- Remark: the proof of $\Leftarrow$ contains a non-constructive case distinction that cannot be removed !!

2.3.4 Proof that 2. $\Leftrightarrow$ 3.

We sketch the proof of 2. $\Leftrightarrow$ 3.

- $\Rightarrow$. Assume that x has a binary expansion that is computed by some Turing machine. Let q_n be the rational number that one obtains by replacing in this expansion all digits from the $(n+2)$th digit after the binary point by zeros. Then sequence $(q_n)_{n \in \mathbb{N}}$ is a computable sequence of rational numbers that converges rapidly to x.
- $\Leftarrow$.
  1. Case x happens to be of the form: integer divided by a power of two. Then it has a binary expansion in which only finitely many digits are different from zero.
  Obviously, such a binary expansion can be produced by a Turing machine.

2. Case x not of this form. Then x has a uniquely determined binary expansion. Let $(q_n)_{n\in\mathbb{N}}$ be a computable sequence of rational numbers converging rapidly to x.

   We which to show that one can compute the binary expansion of x:

   – How to determine the binary expansion in front of the binary point:

     * since x is not an integer, there is an i such that $[q_i - 2^{-i}, q_i + 2^{-i}]$ does not contain an integer: by computing sufficiently many q_i for $i = 0, 1, 2, \ldots$, one can find such a i.
     * Then the binary expansion in front of the binary point of x is equal to the the binary expansion in front of the binary point of q_i.

   – How to determine the binary digit number $n + 1$ in the binary expansion of x, assuming one knows the binary expansion in front of the binary point, and all digits 1, 2, ..., n after the binary point:

     * since x is not an integer divided by 2^{n+1}, there is an i sich that $[q_i - 2^{-i}, q_i + 2^{-i}]$ does not contain an integer divided by 2^{n+1}: by computing sufficiently many q_i for $i = 0, 1, 2, \ldots$, one can find such a i.
     * Then the binary expansion x and of q_i are identical in front of the binary point, and for digits 1, 2, ..., n, $n+1$ after the binary point.

- Remark: the proof of $\Leftarrow$ contains a non-constructive case distinction that cannot be removed !!

2.4 The Field of Computable Reals

Theorem 2 (Rice54) *The set $\mathbb{R}_c$ of computable reals is a real algebraically closed field.*

It means that it is a subfield of the field $\mathbb{R}$ of real numbers, and it contains the real zeros of any polynomials whose coefficients are computable real numbers.

- The field $\mathbb{R}_c$ contains all particular real numbers we ever met in analysis (e.g. $\sqrt{2}, \pi, e$, etc.).
- $\mathbb{Q} \subset \mathbb{A} \subset \mathbb{R}_c \subset \mathbb{R}$ ($\mathbb{A}$ denotes the set of algebraic numbers).
- $\mathbb{R}_c$ is countable.
- $\mathbb{R}_c$ is not complete.

Remark 1 *In the Russian school (ex. Markov, Sanin, Kushner, Aberth) one focus on computability of functions $f : \mathbb{R}_c \to \mathbb{R}_c$.*

Proposition 1 *The set of computable real numbers is countably infinite.*

Proof: It is infinite since it contains all rational numbers. It is countable since for every computable real numbers, there is a Turing machine that computes its binary expansion, and there are countably many Turing machines. □

Proposition 2 *The set of computable real numbers is not complete.*

Proof: Any real numbers, also a non-computable one, is the limit of sequence of rational numbers, hence the limit of a sequence of computable real numbers. □

2.5 Weaker Notions of Computable Reals

2.5.1 Left/Right computable real numbers

Definition 2 *A real number x is* left-computable *if it satisfies one (and then all) of the conditions of the following proposition. A real number x is* right-computable *if $-x$ is right computable.*

Proposition 3 *For a real number x, the following are equivalent:*

- *There exists a computable strictly increasing sequence of rational numbers with limit x.*
- *There exists a computable nondecreasing sequence of rational numbers with limit x.*
- *The set $\{q \in \mathbb{Q} | q < x\}$ is a recursively enumerable subset of $\mathbb{Q}$.*

2.5.2 Relations with Previous Notion

Proposition 4 *A real number is computable iff it is left computable and right computable.*

2.6 Teaser

Theorem 3 *The function $f : \mathbb{R} \to \mathbb{R}$, $f(x) = x + x + x$ is not (ρ_{10}, ρ_{10})-computable.*

Proof:[of Theorem 4] Assume for the sake of contradiction that there is a Turing machine that given a ρ_{10}-name of an arbitrary real x, produces a ρ_{10}-name of $3x$.

Feed $0.3333\ldots$ to the machine. The machine has to produce either $1.000000\ldots$ or $0.999999\ldots$.

Assume it produces $0.999999\ldots$: the machine will write the first 0 after finitely many steps, hence after reading only a finite number of symbols of its input tape, say after reading at most the prefix 0.3^n, for some $n \in \mathbb{N}$. But, then the output of the machine on input $0.3^n4444\ldots$ should not start by 0 (because $3 * \rho_{10}(0.3^n4444\ldots) > 1$,) whereas the machine will output 0. Contradiction.

If it produces $1.00000\ldots$, one reaches at a similar contradiction.

□

Chapter 3

The Notion of Computable Function over the Reals

3.1 Computable Functions over the Integers, Rationals, etc.

3.1.1 Back to Computability Theory

We extend the notions of Section 2.2.2.

We will also consider computability notions on $\mathbb{N}^n$ and $\mathbb{Q}^m$ for $n, m > 0$ as basic, in the same spirit as in previous chapter and in particular in the spirit of Section 2.2.2.

For completeness sake, and in order to avoid misunderstandings, we formally introduce computability of functions on or between these sets via the Turing machine model.

Via the Turing machine model one defines computability on functions on Σ^*, where Σ is a finite alphabet.

To talk about computability over $\mathbb{N}^n$ and $\mathbb{Q}^m$, we need to represent elements of these spaces by strings, that is to say, to fix notations of these spaces.

3.1.2 Notations

Formally:

Definition 3 *A* notation *of a set X, is a surjective function $\nu : \Sigma^* \to X$, where Σ^* is the set of finite words over alphabet Σ.*

3.1.3 Fixing Notations for Usual Objects

Integers

For $X = \mathbb{N}$, we can fix $\nu_{\mathbb{N}} : \Sigma^* \to \mathbb{N}$ to be the usual binary notation of natural numbers.

Pairs, Triples, etc.

Fix a bijection $< ., . >$ from $\mathbb{N}^2$ to $\mathbb{N}$: consider for example

$$< x, y >= \frac{(x+y)(x+y+1)}{2} + y.$$

This yields a bijection $< ., ., \ldots, . >$ from $\mathbb{N}^{k+1} \to \mathbb{N}$ for $k \geq 1$ defined recursively by

$$< x_1, x_2, \cdots, x_{k+1} >=<< x_1, x_2, \cdots, x_k >, x_{k+1} > .$$

Tuple of integers

For $X = \mathbb{N}^n$, we can fix

$$\nu_{\mathbb{N}^n}(w) = (x_1, \cdots, x_n) \text{ iff } \nu_{\mathbb{N}}(w) =< x_1, \cdots, x_n > .$$

Rational Numbers

Consider surjection $\nu_Q : \mathbb{N} \to \mathbb{Q}$ defined by

$$\nu_Q(< i, j, k >) = \frac{i-j}{k+1}.$$

For $X = \mathbb{Q}^n$ (possibly with $n = 1$), we can fix

$$\nu_{\mathbb{Q}^n}(w) = (q_1, \cdots, q_n) \text{ iff } q_i = \nu_Q(\nu_{\mathbb{N}^n}(w)_i).$$

Words

Finally, for $X = \Sigma^*$, consider ν_{Σ^*} to be the identity.

3.1.4 Computable Function over Classical Discrete Spaces

Definition 4 *Consider X and Y to be any of the spaces Σ^*, $\mathbb{N}^n$, or $\mathbb{Q}^m$ for some $n, m \geq 1$.*

A function $f : X \to Y$ is said computable if there exists a Turing machine that on input $w \in \Sigma^*$, never stops if $w \notin dom(f\nu_X)$, and that stops after finitely many steps with some output v such that

$$\nu_Y(v) = f(\nu_X(w))$$

if $w \in dom(f\nu_X)$.

We will also use the notions "decidable" and "computably enumerable" for subsets of Σ^*, $\mathbb{N}^n$, or $\mathbb{Q}^m$ in the usual sense.

3.2 Computability over infinite words

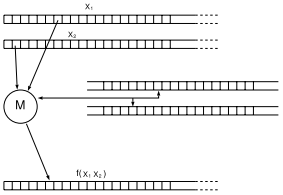
3.2.1 Why talking about infinite words ?

We want to formalize the notion of computable function for a function $f : \mathbb{R} \to \mathbb{R}$.

We can not build a notation for $\mathbb{R}$, as $\mathbb{R}$ is not countable.

We come to the notion of representation, where elements are represented by infinite words, instead of finite words.

3.2.2 Formalizing Computability over Infinite words



Definition 5 *A Turing machine M computes a function $F : \Sigma^\omega \to \Sigma^\omega$, if on input $p \in \Sigma^\omega$, the following holds true:*

- *if $p \in dom(F)$, then M computes infinitely long, and in the long run, it writes the infinite function value $F(p)$ on the write-only output tape.*
- *if $p \notin dom(F)$, then M does not produce an infinite output.*

3.3 Computable Functions over the Reals

3.3.1 Representations

Definition 6 *A* representation *of a set X is a surjective function $\delta : \Sigma^\omega \to X$, where Σ^ω denotes the infinite words over alphabet Σ.*

When $\delta(p) = x$, p is called a name *of x.*

3.3.2 Particular Representations

Decimal Representation

The usual decimal representation ρ_{10} is the representation such that $\rho_{10}(1.414\ldots) = \sqrt{2}$.

Cauchy Representation

Definition 7 (Cauchy representation) *The* Cauchy representation $\rho : \Sigma^\omega \to \mathbb{R}$ *of the real numbers is defined by*

$$\rho(w_0\#w_1\#w_2\ldots) = x \text{ iff } |x - \nu_{\mathbb{Q}}(w_i)| < 2^{-i}$$

for all $i \in \mathbb{N}$.

In other words, a real number x *is represented by a (word coding a) sequence of rational numbers converging rapidly to* x.

Left, Right Representation

Definition 8 (Representation $\rho_<$) *The representation* $\rho_< : \Sigma^\omega \to \mathbb{R}$ *is defined by*

$$\rho_<(w_0\#w_1\#w_2\dots) = x \text{ iff } \{\nu_{\mathbb{Q}}(w_i)|i \in \mathbb{N}\} = \{q \in \mathbb{Q}|q < x\}.$$

Definition 9 (Representation $\rho_>$) *The representation* $\rho_> : \Sigma^\omega \to \mathbb{R}$ *is defined by*

$$\rho_>(w_0\#w_1\#w_2\dots) = x \text{ iff } \{\nu_{\mathbb{Q}}(w_i)|i \in \mathbb{N}\} = \{q \in \mathbb{Q}|q > x\}.$$

3.3.3 Combining representations

Definition 10

- *Let* $\delta : \Sigma^\omega \to X$ *and* $\delta' : \Sigma^\omega \to X'$ *be representations of* X *and* X' *respectively.*

  Then one can consider a representation of $X \times X'$ *defined by*

  $$[\delta, \delta'](< p, q >) = (x, y) \text{ iff } \delta(p) = x \text{ and } \delta'(q) = y,$$

  where $< p, q >= p(0)q(0)p(1)q(1)\dots$.

- *For* $k \geq 1$, *the representation* δ^k *of* X^k *is defined recursively by* $\delta^1 = \delta$, $\delta^{k+1} = [\delta^k, \delta]$.

- *Sometimes, we will need to combine notations and representations: given* $\nu : \Sigma^* \to X$ *and* $\delta : \Sigma^\omega \to X'$ *be notation of* X *and a representation of* X', *then one can consider a representation of* $X \times X'$ *defined by*

  $$[\nu, \delta](0a_10a_20\dots0a_n1p) = (x, x') \text{ iff } \delta(p) = x' \text{ and } \nu(a_1a_2\dots a_n) = x.$$

3.3.4 Computable functions

Definition 11 *Let* $f : X \to Y$ *be some function, and let* $\delta_X : \Sigma^\omega \to X$ *and* $\delta_Y : \Sigma^\omega \to Y$ *be representations.*

Function f *is said to be* (δ_X, δ_Y)*-computable if there exists a computable function* $F : \Sigma^\omega \to \Sigma^\omega$ *such that*

$$\delta_Y F(p) = f(\delta_X(p)) \text{ for all } p \in dom(f\delta_X).$$

(can be seen as a commutative diagram).

3.3.5 Convention/Definition

Definition 12 *A function* $f : \mathbb{R}^k \to \mathbb{R}$ *is called* computable *if it is* (ρ^k, ρ)*-computable.*

3.3.6 Remark

Definition 13 *Let δ be a representation of X. A point $x \in X$ is said δ-computable if there exists a computable $p \in \Sigma^\omega$ (i.e. a p produced as the output of Turing machines working over infinite words without input) such that $\delta(p) = x$.*

Corollary 1 *The computable real numbers coincide with the ρ-computable real numbers.*

3.3.7 Motivation: Multiplication by 3

We indeed consider the Cauchy's representation instead of the (possibly) more natural (usual) decimal representation, because of the following observation:

Theorem 4 *The function $f : \mathbb{R} \to \mathbb{R}$, $f(x) = x + x + x$ is not (ρ_{10}, ρ_{10})-computable.*

Corollary 2 *Addition, multiplication are not computable with respect to the decimal representation.*

Whereas

Theorem 5 *The function $f : \mathbb{R} \to \mathbb{R}$, $f(x) = x + x + x$ is computable (that is to say (ρ, ρ)-computable.*

Proof:[of Theorem 4] Assume for the sake of contradiction that there is a Turing machine that given a ρ_{10}-name of an arbitrary real x, produces a ρ_{10}-name of $3x$.

Feed $0.3333\ldots$ to the machine. The machine has to produce either $1.000000\ldots$ or $0.999999\ldots$.

Assume it produces $0.999999\ldots$: the machine will write the first 0 after finitely many steps, hence after reading only a finite number of symbols of its input tape, say after reading at most the prefix 0.3^n, for some $n \in \mathbb{N}$. But, then the output of the machine on input $0.3^n4444\ldots$ should not start by 0 (because $3 * \rho_{10}(0.3^n4444\ldots) > 1$,) whereas the machine will output 0. Contradiction.

If it produces $1.00000\ldots$, one reaches at a similar contradiction.

□

3.3.8 Philosophical Remark

One may think that the decimal representation is more natural. However, as basic functions like addition or multiplications are not computable with respect to this representation, one then can think:

- either the Turing machine model is not appropriate for real numbers in principle.
- or the model has to be modified such that addition or multiplication become computable

- This can be achieved by allowing two-way output tapes. However, then the model is not closed under composition and the output after finite time would be completely useless
- Alternatively, one must replace the decimal representation by a more suitable representation. This is historically what has been done in Recursive Analysis, following the correction of Turing in 1937 about his paper from 1936.

3.4 Computability of Common functions

3.4.1 Computable Functions

Theorem 6 *The following functions are computable*

- *The arithmetic operations* $+, -, ., :\subset \mathbb{R} \times \mathbb{R} \to \mathbb{R}$.
- *The absolute value function abs* $: \mathbb{R} \to \mathbb{R}, x \to |x|$.
- *The functions* $\min, \max : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$.
- *The constant functions* $\mathbb{R} \to \mathbb{R}$, $x \to c$ *with* $c \in \mathbb{R}_c$.
- *The projections* $pr_i : \mathbb{R}^n \to \mathbb{R}, (x_1, \cdots, x_n) \to x_i$.
- *All polynomials* $p : \mathbb{R}^n \to \mathbb{R}$ *with computable coefficients.*
- *The exponential function and the trigonometric functions* $\exp, \sin, \cos : \mathbb{R} \to \mathbb{R}$.
- *The square root function* $\sqrt{}$ *and the logarithm function* $\log$ *on their natural domains.*

Proof: We sketch the proof that addition $f : \mathbb{R}^2 \to \mathbb{R}$ is computable: given two sequences $(q_n)_{n\in\mathbb{N}}$ and $(r_n)_{n\in\mathbb{N}}$ of rational numbers that rapidly converge to x and y respectively, we can compute the sequence $(p_n)_{n\in\mathbb{N}}$ defined by $p_n = q_{n+1} + r_{n+1}$. This sequence converges rapidly to $x + y$:

$$|x + y - p_n| \leq |x - q_{n+1}| + |y - r_{n+1}| \leq 2^{-(n+1)} + 2^{-(n+1)} = 2 * 2^{-n-1} = 2^{-n}.$$

Since addition can be computed on Turing machines, it follows that f is computable as well. □

Remark 2 *Addition requires only a uniform lookahead of one step that does not depend on the input. For functions tat are not uniformly continuous, such as multiplication, the lookahead may depend on the input.*

Proof: We sketch the proof that multiplication $f : \mathbb{R}^2 \to \mathbb{R}$ is computable: given two sequences $(q_n)_{n\in\mathbb{N}}$ and $(r_n)_{n\in\mathbb{N}}$ of rational numbers that rapidly converge to x and y respectively, we can compute the sequence $(p_n)_{n\in\mathbb{N}}$ defined by $p_n = q_{n+m} * r_{n+m}$. This sequence converges rapidly to $x+y$, if m is well chosen.

$$|x * y - p_n| \leq |x| * |y - r_{n+m}| + |r_{n+m}||x - q_{n+m}|.$$

Indeed, take m such that $|x| \leq |q_0|+1 \leq 2^{m-1}$ and $|r_{n+m}| \leq |r_0|+1 \leq 2^{m-1}$. Then

$$|x * y - p_n| \leq 2^{m-1-n-m} + 2^{m-1-n-m} = 2^n.$$

Since multiplication can be computed on Turing machines, it follows that f is computable as well. □

3.4.2 Non-computable basic functions

We will see in next chapter that a computable function over the reals must be continuous. This yields non-computability of functions like the characteristic function of any set, or of the sign function.

Chapter 4

Computability and Continuity

4.1 Computability implies Continuity for computations over infinite words

4.1.1 Cantor's topology

Cantor's space is the metric space Σ^ω over some finite alphabet Σ, endowed with the metric d_C defined by

$$d_C(p,q) = \begin{cases} 2^{-\min\{i|p(i)\neq q(i)\}} \text{ if } p \neq q \\ 0 \text{ otherwise} \end{cases}$$

Let τ_C be the topology induced by the metric d_C. The topological space (Σ^ω, d_C) is also called *Cantor's space*.

The set $\{w\Sigma^\omega | w \in \Sigma^*\}$ is a basis of the topology τ_C: that is to say, any open set is a union of sets of this form.

(graphical representation as a tree).

4.1.2 Computable implies Continuous

Theorem 7 *Each computable function $F : \Sigma^\omega \to \Sigma^\omega$ is continuous with respect to the Cantor topology.*

Proof: Let M be a Turing machine which compute F. Let $p \in dom(F)$ and let v be a prefix of $q = f(p)$.

After a number t of steps machine M with input p has just written v on the output tape. At this time, machine M has read at most a finite prefix w of p: we obtain $F(w\Sigma^\omega) \subset v\Sigma^\omega$.

In other words, F is continuous. □

4.2 Computability implies Continuity for computations over reals

4.2.1 Reasoning about the commutative diagram

This implies that each computable function $f : \mathbb{R} \to \mathbb{R}$ must be continuous (with respect to the usual (=Euclidean) topology).

In short:

- we must have commutative diagram $f\rho = \rho F$, and since ρ is continuous, $f\rho = \rho F$ also.
- since ρ has an open[1] and surjective restriction, it follows that f itself is continuous.

Remark:

- Since
$$\rho(w_0\#w_1\#\ldots\#w_l\#\Sigma^\omega) = \cap_{i=0}^{l} B(\nu_{\mathbb{Q}}(w_i), 2^{-i})$$
ρ indeed maps (relatively) open sets to open sets.
- Since
$$\rho(w\#w\#\ldots\#w\#\Sigma^\omega) = B(\nu_{\mathbb{Q}}(w), 2^{-i})$$
(where there is i times w in the expression) ρ is indeed continuous.

4.2.2 Effective continuity

Let $B(x, \epsilon) := \{y \in \mathbb{R}^n | d(x, y) < \epsilon\}$ be the ball of center x and of radius ϵ.

We fix a total numbering B^n of all open rational balls in $\mathbb{R}^n$ by:

$$B^n(< i_1, \ldots, i_n, j, k >) := B((\nu_{\mathbb{Q}}(i_1), \ldots, \nu_{\mathbb{Q}}(i_n)), \frac{j+1}{k+1}).$$

Definition 14 (Effective continuity) *A function $f : \mathbb{R}^n \to \mathbb{R}$ is* effectively continuous *if there is a recursively enumerable subset $S \subset \mathbb{N}$ with the following two properties.*

1. *For any $< i, j > \in S$, $f(B^n(i)) \subset B^1(j)$;*
2. *For any $x \in dom(f)$, and any $\epsilon > 0$, there is some $< i, j > \in S$ such that $x \in B^n(i)$ such that the radius of $B^1(j)$ is at most as large as ϵ.*

Theorem 8 *A function $f : \mathbb{R}^n \to \mathbb{R}$ is computable iff it is effectively continuous.*

[1] the image of an open set is an open set.

Proof: For simplicity, we will do the proof only for the case $n = 1$.

Sense $\Rightarrow$: Assume that f is effectively continuous. Then given a ρ-name of a point $x \in dom(f)$, one can compute $f(x)$ with arbitrary precision using an enumeration of S. Thus f is computable.

Sense $\Leftarrow$:

- Assume that f is computable. Then a Turing machine given any k and any name p of $x \in dom(f)$, computes a rational q with $f(x) \in B(q, 2^{-k})$.

  Fix k. The machine does so after reading at most a finite prefix of p of the form $w_0\#w_1\#\ldots\#w_l\#$.

  Then $U := \cap_{i=0}^{l} B(\nu_Q(w_i), 2^{-i})$ is an open neighborhood of x, and any point in this neighborhood has a ρ-name starting with the prefix.

  Thus given a name starting with the prefix of any point in U, the machine will produce the same output q. This implies that $f(U) \subset B(q_k, 2^{-k})$.

- This implies that f is continuous (Take any neighborhood V of $y = f(x)$. There must exists some k with $B(q_k, 2^{-k}) \subset V$. Consider respective U, we have $f(U) \subset B(q_k, 2^{-k}) \subset V$.).

- f is even effectively continuous: enumerate systematically all prefixes of ρ-names of real numbers and tests the Turing machines on them, to build a set S.

□

4.2.3 Consequence

This shows that even a function as simple as the sign function ($sign(x) := 0$ if $x < 0$, 1 otherwise) is not computable.

4.3 (Recursive Analysis') Church Turing's Thesis for Real Number Functions

- Discontinuous functions $f : \mathbb{R} \to \mathbb{R}$ cannot be computable.
- They might be "simple to describe" (a computable point in some function space), or computable in some weaker sense (e.g. lower semi-computable)
- but not computably evaluable with arbitrary precision

  (dessin)

(Recursive Analysis') Church Turing's Thesis for Real Number Functions: A function $f : \mathbb{R}^n \to \mathbb{R}$ is computable, if and only if it can be evaluated on a physical computer with arbitrary given precision.

Hence, it must be continuous...

Chapter 5

Computability for functions: oracle view & modulus of continuity

5.1 Using oracles

A dyadic rational number d is a rational number that has a finite binary expansion; that is, $d = m/2^n$ for some integers m, n, $n \geq 0$. Let D be the set of all dyadic rational numbers. The binary expansion of a dyadic rational is its natural finite representation.

We denote by D_n the set of all dyadic rationals d with a representation s of $prec(s) = n$: that is, $D_n = \{m \cdot 2^{-n} | m \in \mathbb{Z}\}$.

The main reason for using set D, instead of set $\mathbb{Q}$, as the base set is that the set D is not only dense in $\mathbb{R}$ but also uniformly dense in R in the following sense: for any integer $n > 0$, dyadic rationals of precision n (more precisely, dyadic rationals having representations s with $prec(s) = n$) are uniformly distributed on the real line. When we define computable real numbers, the base set $\mathbb{Q}$ and the base set D will define the same class of computable real numbers, but when we define complexity of real numbers, the base set D is more natural and easier to use.

Using this notation, we can give, based on the Cauchy sequence representation, a more formal definition of computable real numbers.

Definition 15 *For each real number x, a function $\varphi : \mathbb{N} \to D$ is said to binary converge to x if it satisfies the condition that for all $n \in N$, $prec(\varphi(n)) = n$ and $|\varphi(n) - x| \leq 2^{-n}$. Let CF_x (Cauchy function) denote the set of all functions binary converging to x.*

Theorem 9 *A real x is computable iff CF_x contains a computable function φ.*

Theorem 10 *A real function $f : \mathbb{R} \to \mathbb{R}$ is computable if there is a function-oracle TM M such that for each $x \in \mathbb{R}$ and each $\varphi \in CF_x$, the function ψ computed by M with oracle φ (i.e., $\psi(n) = M^{\varphi}(n)$) is in $CF_{f(x)}$. We say the function f is computable on interval $[a, b]$ if the above condition holds for all $x \in [a, b]$.*

5.2 Modulus of continuity

Definition 16 *Let $f : [a, b] \to \mathbb{R}$ be a continuous function on $[a, b]$. Then, a function $m : \mathbb{N} \to \mathbb{N}$ is said to be a modulus function of f on $[a, b]$ iff for all $n \in \mathbb{N}$ and all $x, y \in [a, b]$, we have*

$$|x - y| \leq 2^{-m(n)} \Rightarrow |f(x) - f(y)| \leq 2^{-n}.$$

The proof of the following theorem uses the Heine-Borel covering theorem that states that each open covering of a closed and bounded set $A \subseteq R$ has a finite subcovering. That is, for any collection C of open sets such that $A \subseteq \bigcup\{S | S \in C\}$, there is a finite collection $\{S_1, ..., S_n\}$ of sets in C such that $A \subseteq \bigcup_{i=1}^{n} S_i$.

Theorem 11 *If $f : [a, b] \to \mathbb{R}$ is computable on $[a, b]$ then f is continuous on $[a, b]$; furthermore, f has a computable modulus function m on $[a, b]$.*

Proof: Let b_x be the standard Cauchy function for x:

$$b_x(n) = BE_x(-1) \cdot (BE_x(0) + \sum_{i=1}^{n} BE_x(i)2^{-i})$$

where BE_x is binary expansion of x

$$x = BE_x(-1) \cdot (BE_x(0) + \sum_{i=1}^{\infty} BE_x(i)2^{-i})$$

Observe that we have: $|y - b_x(n)| < 2^{-n}$ then $|y - b_x(k)| < 2^{-k}$ for all $k \leq n$.

Let M be an oracle TM computing f and n a fixed integer. For each $x \in [a, b]$, consider the computation $M^{b_x}(n + 2)$ of M on input $n + 2$ with oracle b_x. Let $k_x = \max\{k | b_x(k)$ is queried in the computation $M^{b_x}(n + 2)\}$.

Then, we claim that for all $y \in [a, b]$, $|y - b_x(k_x)| < 2^{-k_x}$ implies that $|f(y) - f(x)| \leq 2^{-(n+1)}$. To see this, define a function ψ by $\psi(k) = b_x(k)$ if $k \leq k_x$, and $\psi(k) = b_y(k)$ if $k > k_x$. Then, $\psi \in CF_y$. Note that the computation of $M^{\psi}(n + 2)$ is exactly the same as $M^{b_x}(n + 2)$, because M does not ask about $\psi(k)$ for any $k > k_x$. This implies that $|f(x) - f(y)| \leq |f(x) - M^{b_x}(n + 2)| + |M^{\psi}(n + 2) - f(y)| \leq 2^{-(n+1)}$.

Let $\ell_x = b_x(k_x) - 2^{-k_x}$ and $r_x = b_x(k_x) + 2^{-k_x}$. Then, $y, z \in (\ell_x, r_x) \cap [a, b]$ implies $|f(y) - f(z)| \leq 2^{-(n+1)}$. (Both $f(y)$ and $f(z)$ differ from $M^{b_x}(n + 2)$ by at most $2^{-(n+2)}$.) Consider the open covering$\{(\ell_x, r_x) | x \in [a, b]\}$ for $[a, b]$.

By the Heine-Borel Theorem, there must be a finite covering $\{(\ell_{x_i}, r_{x_i})|i = 1, \ldots, t\}$ for $[a, b]$. Let $m(n) = \max\{k_{x_i}|i = 1, \ldots, t\}$. We claim that for any $y, z \in [a, b]$, $|y - z| \leq 2^{-m(n)}$ implies $|f(y) - f(z)| \leq 2^{-n}$. This can be verified as follows. If $|y - z| \leq 2^{-m(n)}$ and $y < z$, then either $\ell_{x_i} < y < z < r_{x_i}$ for some $i = 1, \ldots, t$,or $\ell_{x_i} < y \leq \ell_{x_j} < r_{x_i} \leq z < r_{x_j}$ for some $i, j = 1, \ldots, t$. In the first case, $|f(y) - f(z)| \leq 2^{-(n+1)}$. In the second case, $|f(y) - f(z)| \leq |f(y) - f(u)| + |f(u) - f(z)| \leq 2^{-n}$, where u is a real number in (ℓ_{x_j}, r_{x_i}). This shows that the function $m(n)$ is a modulus function for f.

Finally we check that m is a computable function. Note that in the above proof, for each $i = 1, \ldots, t$, if we let $d_i = b_{x_i}(k_{x_i}) = (\ell_{x_i} + r_{x_i})/2$, then $d_i \in D$ and the computation of machine M on input $n+2$ with oracle b_{d_i} is exactly the same as that with oracle b_{x_i}. In other words, we have $\ell_{d_i} = \ell_{x_i}$ and $r_{d_i} = r_{x_i}$. It implies that $m(n)$ can be computed as the maximum of k_{d_i}, $i = 1, \ldots, t$. This suggests the following algorithm for $m(n)$:

- for $i = 1$ to ∞ do
  - if $[a, b] \subseteq \bigcup_{d_i \in D_i \cap [a,b]} (\ell_d, r_d)$
    - then halt and output $m(n) = \max\{k_d | d \in D_i \cap [a, b]\}$.

By the above discussion, there exists an integer i such that $\{(\ell_d, r_d)|d \in D_i \cap [a, b]\}$ covers $[a, b]$. So, this algorithm always halts. Furthermore, when it halts in the ith iteration with the output $m(n)$, $\{(\ell_d, r_d)|d \in D_i \cap [a, b]\}$ must be a covering for $[a, b]$. As proved above, this implies that for all $y, z \in [a, b]$, $|y - z| \leq 2^{-m(n)} \Rightarrow |f(y) - f(z)| \leq 2^{-n}$. So the algorithm computes a computable modulus function for f. □

The above gives us immediately our first characterization of computable real functions.

Corollary 3 *A real function $f : [a, b] \to \mathbb{R}$ is computable iff there exist two computable functions $m : \mathbb{N} \to \mathbb{N}$ and $\psi : (D \cap [a, b]) \times \mathbb{N} \to D$ such that*

- *m is a modulus function for f*
- *and for all $d \in D \cap [a, b]$ and all $n \in \mathbb{N}$, $|\psi(d, n) - f(d)| \leq 2^{-n}$.*

Proof: The "only if" part follows immediately from Theorem 11. For the "if" part, assume that m and ψ satisfy 1. and 2. An oracle TM M computing f works as follows: On input n and with oracle φ , query for $d = \varphi(m(n + 1))$ and output $\psi(d, n + 1)$. Note that if $\varphi \in CF_x$ then $|d - x| \leq 2^{-m(n+1)}$ and so $|f(d) - f(x)| \leq 2^{-(n+1)}$. Thus $|f(x) - \psi(d, n + 1)| \leq 2^{-n}$. □

The second characterization is to use piecewise linear functions to approximate computable real functions. Again the characterization is presented for functions defined on a closed interval. For simplicity, we only consider functions defined on $[0, 1]$.

Theorem 12 *A real function f defined on $[0, 1]$ is computable iff there exist a sequence (f_n) of piecewise linear functions on $[0, 1]$ and a computable function m such that*

- *(simple piecewise linearity) for each $n \in \mathbb{N}$, the breakpoints of f_n are in $D_{m(n)}$,and for each $d \in D_{m(n)} \cap [0,1]$, $f_n(d) \in D$;*
- *(uniform modulus) for each $n \in \mathbb{N}$ and each $d \in D_{m(n)} \cap [0,1)$, $|f_n(d) - f_n(d + 2^{-m(n)})| \leq 2^{-n}$;*
- *(uniform convergence) for each $n \in \mathbb{N}$ and each $x \in [0,1]$, $|f_n(x) - f(x)| \leq 2^{-n}$; and*
- *(uniform computability) the function $\psi : (D \cap [0,1]) \times \mathbb{N} \to D$ defined by*

$$psi(d,n) = \begin{cases} f_n(d) & \text{if } d \in D_{m(n)} \\ 0 & \text{otherwise} \end{cases}$$

is computable.

Proof: First assume that there exist a sequence (f_n) of piecewise linear functions on $[0,1]$ and a computable function m satisfying conditions 1. to 4.. For each $x \in [0,1]$, find $d \in D_{m(n+1)}$ such that $|d - x| \leq 2^{-m(n+1)}$. Then, we can approximate $f(x)$ by $\psi(d, n+1)$ within an error 2^{-n}.

Conversely, assume that f is computable on $[0,1]$ by an oracle TM M. Let m_1 be a computable modulus function for f on $[0,1]$, as shown in Theorem 11. Let $m(n) = m_1(n+3)$. For each $n \in \mathbb{N}$, define a piecewise linear function f_n on $[0,1]$ with breakpoints in $D_{m(n)}$ by $f_n(d) = M^{b_d}(n+3)$ for all $d \in D_{m(n)}$. Then (f_n) obviously satisfies conditions 1. and 4.

To verify condition 2., let $d \in D_{m(n)} \cap [0,1)$ and $e = d + 2^{-m(n)}$. Then,

$$\begin{aligned} |f_n(d) - f_n(e)| &= |M^{b_d}(n+3) - M^{b_e}(n+3)| \\ &\leq |M^{b_d}(n+3) - f(d)| + |f(d) - f(e)| + |f(e) - M^{b_e}(n+3)| \\ &\leq 3 \cdot 2^{-(n+3)} \\ &< 2^{-(n+1)} \end{aligned}$$

For condition 3., consider $x \in [0,1]$ and $d = b_x(m(n))$. Then,

$$\begin{aligned} |f_n(x) - f(x)| &\leq |f_n(x) - f_n(d)| + |f_n(d) - f(d)| + |f(d) - f(x)| \\ &< 2^{-(n+1)} + 2^{-(n+3)} + 2^{-(n+3)} \\ &< 2^{-n}. \end{aligned}$$

This completes the proof. □

Although the above characterizations are stated for functions with a compact domain, they can easily be extended to functions defined on $\mathbb{R}$ by extending the notion of uniform modulus of continuity to a local modulus of continuity.

We state such an extension of previus corollary:

Corollary 4 *A function $f : \mathbb{R} \to \mathbb{R}$ is computable iff there exist two recursive functions $m : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ and $\psi : D \times \mathbb{N} \to D$ such that*

- *for all* $k, n \in \mathbb{N}$ *and all* $x, y \in [-k, k]$, $|x - y| \leq 2^{-m(k,n)}$ *implies* $|f(x) - f(y)| \leq 2^{-n}$, *and*
- *for all* $d \in D$ *and all* $n \in \mathbb{N}$, $|\psi(d, n) - f(d)| \leq 2^{-n}$.

Chapter 6

Computability for Subsets

6.1 First considerations

Over $\mathbb{N}$, not only computable functions are of fundamental important, but also effective notions for sets, like the notions of computable (= recursive, decidable) set, and recursively enumerable (=computably enumerable, semi-decidable) sets.

Over $\mathbb{N}$, a subset $A \subset \mathbb{N}^n$ is decidable if its characteristic function $\chi_A : \mathbb{N} \to \mathbb{N}$ is computable.

Let us first to translate this idea to subsets of $\mathbb{R}^n$:

Definition 17 *Let X be a set, and δ_X be a representation of X.*

A subset $A \subset X$ is said to δ_X-decidable if its characteristic function $\chi_A : X \to \mathbb{R}$ is (δ_X, ρ)-computable.

The (ρ^n, ρ)-decidable subsets of $\mathbb{R}^n$ will be called decidable

Proposition 5 *The only decidable subsets of $\mathbb{R}^n$ are $\emptyset$ and $\mathbb{R}^n$.*

Proof: The characteristic functions of $\emptyset$ and $\mathbb{R}^n$ are the constant functions 0 and 1 and are computable.

The characteristic function of any other subset is discontinuous and hence non-computable. □

Is the problem with the ρ-representation?

Proposition 6 *There is no representation $\delta : \Sigma^\omega \to \mathbb{R}$ of the real numbers such that any of the tests $=, <, \leq$ are decidable with respect to δ: that is to say, such that any of the following subsets of $\mathbb{R}^2$ is δ^2-decidable:*

$$\{(x, y)| \in \mathbb{R}^2 | x = y\}$$

$$\{(x, y)| \in \mathbb{R}^2 | x < y\}$$

$$\{(x, y)| \in \mathbb{R}^2 | x \leq y\}$$

Proof: We do the proof only for $=$.

Assume that there is a representation δ such that $=$ is decidable. Consider an arbitrary real number x and a δ-name p of x. On input $< p, p >$ after a finite time, the Turing machine outputs 1. It does so after reading at most a finite prefix of its input, say at most $2n$ symbols of $< p, p >$. Then for any name q that starts with $u := p(0)p(1)\dots p(n)$, the machine on input $< p, q >$ will output 1.

This implies that $\delta(p) = \delta(q) = x$ for any δ-name q starting with the prefix u. Since there are countably many strings in Σ^*, and δ is surjective, this implies that $\mathbb{R}$ is countable.

Contradiction. □

6.2 Better considerations: Defining computable subsets

6.2.1 Definitions

Consider a non-empty closed subset $A \subset \mathbb{R}^n$. The distance function $d_A : \mathbb{R}^n \to \mathbb{R}$ defined by

$$d_A(x) := \inf_{y \in A} d(x, y),$$

can be considered as a "smooth" version of the characteristic function of A.

(dessin).

Definition 18 *A closed subset $A \subset \mathbb{R}^n$ is said* computable *or* recursive *if either it is empty or d_A is computable.*

Definition 19 *An open subset $A \subset \mathbb{R}^n$ is said* computable *or* recursive *if its complement (which is a closed set) is computable.*

6.2.2 This extends the notions over $\mathbb{N}^n$

Proposition 7 *A subset $A \subset \mathbb{N}^m$ is decidable if and only if it is computable when considered as a closed subset of $\mathbb{R}^m$.*

Proof: Idea for $\Leftarrow$ (for $m = 1$).

Observe that for any integer $n \in \mathbb{N}$, $d_A(n)$ is 0 if $n \in A$, or ≥ 1 for $n \notin A$.

Given $n \in \mathbb{N}$, if for a k we know that

1. $d_A(n) < 1 - 2^{-k}$ then $n \in A$.
2. $d_A(n) > 1/2 + 2^{-k}$ then $n \in \mathbb{N} - A$

Increase k until one of the two conditions above hold. □

Examples of computable subsets.

1. $\{x\}$ is computable iff x is a computable point.

2.
$$\{(x,y) \in \mathbb{R}^2 | x = y\}$$
$$\{(x,y) \in \mathbb{R}^2 | x < y\}$$
$$\{(x,y) \in \mathbb{R}^2 | x \leq y\}$$
are computable

3. The open ball $B(x,\epsilon)$ and the corresponding closed ball are computable, when x and ϵ are computable.

4. $[a,b]$ is computable when a and b are.

5. For any total function $f : \mathbb{R}^n \to \mathbb{R}$ the graph $\{(x_1,\ldots,x_n,y)|y = f(x_1,\ldots,x_n)$ is computable when f is computable.

6.2.3 Natural interpretation

The idea is that a closed set is computable if one can plot pixel of it at arbitrary precision.

(dessin).

(Computability on $\mathbb{Z}$ can be reduced to computability on $\mathbb{N}$ by $\nu(2n) := n$, and $\nu(2n+1) = -(n+1)$).

Proposition 8 *For a closed subset $A \subset \mathbb{R}^k$, the following conditions are equivalent.*

1. *A is computable.*

2. *There exists a computable function $f : \mathbb{N} \times \mathbb{Z}^k \to \mathbb{N}$ with $range(f) \subset \{0,1\}$ such that for all $n \in \mathbb{N}$, $z \in \mathbb{Z}^k$,*
$$f(n,z) = \begin{cases} 1 & \text{if } d_A(\frac{z}{2^n}) < 2^{-n} \\ 0 & \text{if } d_A(\frac{z}{2^n}) > 2.2^{-n} \\ 0 \text{ or } 1 & \text{otherwise} \end{cases}$$

6.3 Better considerations: Defining recursively enumerable subsets / Co-re closed sets

6.3.1 Back to classical computability

For subsets of $\mathbb{N}^n$ the following is well-known:

Proposition 9 *The following are equivalent:*

1. *$A \subset \mathbb{N}$ is recursively enumerable.*

2. *there is some computable $f : \mathbb{N} \to \mathbb{N}$ with $dom(f) = A$.*

3. *A is empty or there is some total computable function $f : \mathbb{N} \to \mathbb{N}$ such that $A = range(f)$.*

co-recusively enumerable means "whose complement is" recursively-enumerable.

The idea is to generalize these to subsets of $\mathbb{R}^n$. The generalization of condition 2. leads to point 1. in the proposition below. The generalization of condition 3. leads to points 2. and 3. in proposition below.

6.3.2 For subsets of $\mathbb{R}^n$

Definition 20 *Let X be a set with representation $\delta : \Sigma^\omega \to X$. A set $U \subset X$ is called δ-recursively enumerable (= δ-computably enumerable) if a Turing machine, given any arbitrary δ-name p of an arbitrary element $x \in X$, halts after finitely many steps if and only if $x \in U$.*

Proposition 10 *Let $U \subset \mathbb{R}^n$ be open and let $A := \mathbb{R}^n - U$ be its (closed) complement.*

The following conditions are equivalent:

1. *U is ρ^n-computably enumerable*

2. *$U = \bigcup_{(q,\epsilon)\in S} B(q, \epsilon)$ for some recursively enumerable set $S \subset \mathbb{Q}^n \times \mathbb{Q}_+$ (here: $\mathbb{Q}_+ = \mathbb{Q} \cap \{q > 0\}$)*

3. *The set $\{(q, \epsilon) \in \mathbb{Q}^n \times \mathbb{Q}_+ | \overline{B}(q, \epsilon) \subset U\}$ is recursively enumerable*

4. *$A = f^{-1}(\{0\})$ for some total computable function $f : \mathbb{R}^n \to \mathbb{R}$*

5. *The function $\chi_A : \mathbb{R}^n \to \mathbb{R}$ is lower semicomputable: i.e. it is $(\rho^n, \rho_<)$-computable.*

6. *Either $A = \emptyset$ or the function $d_A : \mathbb{R}^n \to \mathbb{R}$ is lower semicomputable.*

Proof: 6. $\Rightarrow$ 3: Follows from $d_A(x) > \epsilon$ iff $A \cap \overline{B}(x, \epsilon) = \emptyset$.
2. $\Rightarrow$ 4: if $U = \bigcup_{(x_i,r_i)} B(x_i, r_i)$ then

$$f(x) := \sum_{i=0}^{\infty} \frac{\max(0, r_i - d(x_i, x))}{r_i} 2^{-i-1}$$

defines a computable function $f : \mathbb{R}^n \to \mathbb{R}$ with $A = f^{-1}(\{0\})$. □

Definition 21 *If one (an then all) of the conditions above are true, U is said* computably enumerable open*, and A is said* co-computably enumerable closed*.*

6.3.3 Intuition/Digression

Let $A \subset \mathbb{R}^n$ be a closed set. Then A is co-recursively enumerable if

$$\{(q, \epsilon) \in \mathbb{Q}^n \times \mathbb{Q}_+ | \overline{B}(q, \epsilon) \cap A = \emptyset\}$$

is recursively enumerable.

(dessin).

This corresponds to "negative information" on the set A.

Definition 22 *The upper Fell topology on $\mathcal{A} := \{A \subset \mathbb{R}^n | A$ is closed $\}$ is generated by the subbase which consists of the sets $\{A \in \mathcal{A} | A \cap K = \emptyset\}$ for all compact sets $K \subset \mathbb{R}^n$.*

6.4 Better considerations: Defining recursively enumerable subsets / Re closed sets

There is another generalization of computable enumerability:

Proposition 11 *Let $A \subset \mathbb{R}^n$ be closed.*
The following conditions are equivalent:

1. *The set $\{(q, \epsilon) \in \mathbb{Q}^n \times \mathbb{Q}_+ | B(q, \epsilon) \cap A \neq \emptyset\}$ is recursively enumerable.*
2. *Either $A = \emptyset$ or there is a computable sequence $(x_i)_{i \in \mathbb{N}}$ of points $x_i \in \mathbb{R}^n$ such that A is the closure of the set $\{x_i | i \in \mathbb{N}\}$.*
3. *Either $A = \emptyset$ or the function $d_A : \mathbb{R}^n \to \mathbb{R}$ is upper semicomputable.*

Definition 23 *If one (an then all) of the conditions above are true, A is said* computably enumerable closed *and U is said* co-computably enumerable open.

Proposition 12 *1. A subset $A \subset \mathbb{N}^n$ is computably enumerable (considered as a subset of $\mathbb{N}^n$) if and only if it is computably enumerable when considered as a closed subset of $\mathbb{R}^n$.*

2. A subset $A \subset \mathbb{N}^n$ is co-computably enumerable (considered as a subset of $\mathbb{N}^n$) if and only if it is co-computably enumerable when considered as a closed subset of $\mathbb{R}^n$.

Proposition 13 *An open or closed subset of $\mathbb{R}^n$ is computable iff it is computably enumerable and co-computably enumerable.*

Proof: Follows from the fact that lower semi-computable + upper-semicomputable = computable. □

6.4.1 Intuition/Digression

Let $A \subset \mathbb{R}^n$ be a closed set. Then A is recursively enumerable if

$$\{(q, \epsilon) \in \mathbb{Q}^n \times \mathbb{Q}_+ | B(q, \epsilon) \cap A \neq \emptyset\}$$

is recursively enumerable.

(dessin).

This corresponds to "positive information" on the set A.

Definition 24 *The lower Fell topology on $\mathcal{A} := \{A \subset \mathbb{R}^n | A \text{ is closed } \}$ is generated by the subbase which consists of the sets $\{A \in \mathcal{A} | A \cap U \neq \emptyset\}$ for all open $U \subset \mathbb{R}^n$.*

6.5 Closure properties

Proposition 14 *Let A and B be closed subset of $\mathbb{R}^n$.*

- *If A and B are co-ce closed, so are $A \cup B$ and $A \cap B$.*
- *If A and B are ce closed, so are $A \cup B$.*
- *If A and B are computable, so is $A \cup B$.*

Proof:

1. Let $f, g : \mathbb{R}^n \to \mathbb{R}$ be computable functions such that $A = f^{-1}(\{0\})$ and $B = g^{-1}(\{0\})$. Then $A \cup B = (f * g) - 1(\{0\})$ and $A \cap B = (|f| + |g|)^{-1}(\{0\})$, and $f * g$ and $|f| + |g|$ are computable functions.
2. A closed set is c.e. closed iff the set of open rational balls intersecting the set can be enumerated effectively. Now note that any open (rational) ball intersect $A \cup B$ iff it intersects at least one of A and B.
3. Follows from other statements.

□

6.6 Some open questions

Recall the Mandelbrot set M: $M = \{c \in \mathbb{C} |$ all numbers of the sequence $(z_i)_{i \in \mathbb{N}}$ defined by $z_0 := 0$, and $z_{i+1} := z_i^2 + c$ satisfy $|z_i| \leq 2\}$.

The following questions are open

1. Is the Mandelbrot set computable?
2. Is the interior of the Mandelbrot set c.e. open?

The (famous) hyperbolicity conjecture says that a certain subset of interior of the Mandelbrot set is actually equal to interior of the Mandelbrot set.

Proposition 15 (Hertling 2005) *If the hyperbolicity conjecture is true, then the answer to both questions above is yes.*

Chapter 7

Intermediate Value Theorem

7.1 Introduction

Theorem 13 *For each continuous function $f : [0,1] \to \mathbb{R}$ with $f(0) * f(1) < 0$ there exists a point $x \in [0,1]$ with $f(x) = 0$.*

Does there exist a computable version of this theorem?
In general, we can distinguish two computable forms of such theorems:

1. *non-uniform*: for any suitable computable f, there exists a computable x.
2. *uniform*: given some suitable continuous f, we can compute a zero x.

In the uniform case, we can distinguish two subcases:

1. *functional*: there exists a computable (continuous) function $Z : \mathcal{C}[0,1] \to \mathbb{R}$ such that $Z(f)$ is a zero of f for all $f \in F$.
2. *multi-valued*: there exist a multi-valued computable (continuous) function $Z : \mathcal{C}[0,1] \rightrightarrows \mathbb{R}$ such that for all $f \in F$ there exists an $x \in Z(f)$ and all such x are zeroes of f.

   Here $F \subset \{f \in \mathcal{C}[0,1] | f(0)f(1) < 0\}$.

7.2 Trisection method

Theorem 14 *There exists an algorithm which determines a zero for all given computable (hence continuous) function $f : [0,1] \to \mathbb{R}$ with $f(0) * f(1) < 0$ which have exactly one zero.*

Proof: In the classical bissection method, function values are compared with 0. As we saw that this cannot be done in finite time, we use a *trisection* method.

We start with $a_0 := 0$ and $d_0 := 1$. Now, assume that we have computed two numbers $a_i, d_i \in [0,1]$ with $d_i - a_i = (2/3)^i$ such that $f(a_i) * f(d_i) < 0$ (this is true for $i = 0$).

In parallel, we compute the two values $f(a_i) * f(a_i + 2/3(d_i - a_i))$ and $f(a_i + 1/3(d_i - a_i)) * f(d_i)$ with higher and higher precision until one of them turns out to be smaller than 0 (at least one of them must be smaller than 0). If this is the first one, we set $a_{i+1} := a_i, d_{i+1} := a_i + 2/3(d_i - a_i)$. If this is the second one, we set $a_{i+1} := a_i + 1/3(d_i - a_i), d_{i+1} := d_i$. □

Remark 3 *If we used a bissection method, we would cut $[a_i, d_i]$ in half, and test the sign of $f(z_i)$ in $z_i = (a_i + d_i)/2$: if $f(z_i) \neq 0$, (by considering higher and higher precision), we could ultimately know which case of $f(z_i) > 0$ or $f(z_i) < 0$ holds; the point is that it may happen than $f(z_i) = 0$, and the algorithm would never stop.*

The trick here is that we cannot have simultaneously $f(b_i) = 0$ and $f(c_i) = 0$ for $b_i = a_i + 1/3(d_i - a_i)$ and $c_i = a_i + 2/3(d_i - a_i)$. Hence, the algorithm always terminate.

7.3 Stronger statement

We actually proved a stronger statement: there exists a corresponding computable functional: $Z : \mathcal{C}[0,1] \to \mathbb{R}$.

We need to fix a representation of functions of $\mathcal{C}[0,1]$.

Actually, we can generalize what we did for reals that we represented by Cauchy sequences of Rationals. We will here consider functions as Cauchy sequences of polynomial functions.

In full generality, we are using the notion of computable metric space.

Definition 25 *A triple (X, d, α) is called a* computable metric space *if*

1. *$d : X \times X \to \mathbb{R}$ is a metric on X*
2. *$\alpha : \mathbb{N} \to X$ is a sequence such that the set $\{\alpha(n) | n \in \mathbb{N}\}$ is dense in X*
3. *$d \circ (\alpha \times \alpha) : \mathbb{N}^2 \to \mathbb{R}$ is a computable (double) sequence in $\mathbb{R}$.*

Definition 26 *Let (X, d, α) be a computable metric space. Then we define the Cauchy representation $\delta_X : \Sigma^\omega \to X$ by*

$$\delta_X(01^{n_0+1}01^{n_1+1}01^{n_2+1}\dots) = \lim_{i\to\infty} \alpha(n_i).$$

for any sequence $(n_i)_{i\in\mathbb{N}}$ such that $(\alpha(n_i))_{i\in\mathbb{N}}$ converges rapidly: we say that a sequence $(x_i)_{i\in\mathbb{N}}$ converges rapidly , if $d(x_i, \lim_{n\to\infty} x_n) < 2^{-i}$.

Examples

- $(\mathbb{R}^n, d, \alpha_{\mathbb{R}^n})$ with the Euclidean metric $d(x,y) = \sqrt{\sum_{i=1}^n |x_i - y_i|^2}$ and the standard numbering $\alpha_{\mathbb{R}^n}$ of $\mathbb{Q}^n$ defined by

$$\alpha_{\mathbb{R}^n}(< i_1, \ldots, i_n >) = (\nu_{\mathbb{Q}}(i_1), \ldots, \nu_{\mathbb{Q}}(i_n)).$$

  is a computable metric space.

- $(\mathcal{C}[0,1], d_{\mathcal{C}}, \alpha_{\mathcal{C}})$ is a computable metric space, with

$$d_{\mathcal{C}}(f,g) := ||f - g|| = \sup_{x \in [0,1]} |f(x) - g(x)|.$$

  and some standard numbering $\alpha_{\mathcal{C}}$ of $\mathbb{Q}[x]$, for instance

$$\alpha_{\mathcal{C}}(< k, < i_0, \ldots, i_k >>) = \sum_{j=0}^{k} \nu_{\mathbb{Q}}(i_j) x^j.$$

Notes:

1. The function $(f,x) \to f(x)$ is $([\delta_{\mathcal{C}[0,1]}, \rho^{[0,1]}], \rho)$-computable (where $\rho^{[0,1]}$ is the restriction of ρ to names of real numbers in $[0,1]$)

2. The computable points of $\delta_{\mathcal{C}[0,1]}$ are exactly the computable functions $f : [0,1] \to \mathbb{R}$.

Theorem 15 *There is a Turing machine which, given a $\delta_{\mathcal{C}[0,1]}$-name of some function f with $f(0) * f(1) < 0$ and such that $f^{-1}(\{0\})$ does not contain an interval computes a (ρ-name of) a zero of f.*

Proof: We use a trick similar to the trisection method, with a slight generalization.

First, we note that if f is such a function, and $a < d$ are numbers in $[0,1]$ with $f(a) * f(d) < 0$, then for any $n \in \mathbb{N}$ there are rational numbers b and c with $a \leq b < c \leq d$ with $c - b \leq 2^{-n}$ and $f(b) * f(c) < 0$.

Assume that a $\delta_{\mathcal{C}[0,1]}$ of such a function f is given: similarly to the trisection algorithm, we start with $a_0 := 0$ and $d_0 := 1$, and once two rational numbers a_i and d_i with $a_i < d_i$ and $d_i - a_i \leq 2^{-i}$, and $f(a_i) * f(d_i) < 0$ are found, it searches for two rational numbers a_{i+1}, d_{i+1} with $a_i \leq a_{i+1} < d_{i+1} \leq d_i$ with $d_{i+1} - a_{i+1} \leq 2^{-i-1}$ with $f(a_{i+1}) * f(d_{i+1}) < 0$. □

Remark 4 *Actually the algorithm returns possibly different zeros of f depending on the $\delta_{\mathcal{C}[0,1]}$ name of f given to the algorithm.*

7.4 Unsolvability in the general case

Theorem 16 *There is no general algorithm which determines a zero for any continuous function $f : [0,1] \to \mathbb{R}$ with $f(0) * f(1) < 0$.*

Definition 27 *Assume δ_X is a representation of X, and δ_Y a representation of Y. A multi-valued function $f : X \rightrightarrows Y$ is called (δ_X, δ_Y)-computable if there exists a computable function $F : \Sigma^\omega \to \Sigma^\omega$ such that $\delta_Y F(p) \in f\delta(p)$ for all $p \in dom(f\delta_X)$.*

Analogously, f is called (δ_X, δ_Y)-continuous if there exists a continuous F with the above property.

(dessin).

Actually, the above theorem follows from the following observation:

$Z : \mathcal{C}[0,1] \rightrightarrows \mathbb{R}$ defined by $dom(Z) := \{f \in \mathcal{C}[0,1] | f(0) * f(1) < 0\}$, $Z(f) = f^{-1}(\{0\}$, is not $(\delta_{\mathcal{C}[0,1]}, \rho)$-continuous.

Proof: For the sake of contradiction, assume that Z is $(\delta_{\mathcal{C}[0,1]}, \rho)$-continuous. Consider the continuous piecewise linear function $h_y : [0,1] \to \mathbb{R}$ with breakpoints $(0,-1)$, $(1/3, y)$, $(2/3, y)$, $(1,1)$, for arbitrary $y \in \mathbb{R}$. Given a ρ-name of y, one can compute a $\delta_{\mathcal{C}[0,1]}$-name of h_y. If Z were $(\delta_{\mathcal{C}[0,1]}, \rho)$-continuous, then there would also be a continuous function mapping any ρ-name of y to a ρ-name of a zero h_y.

The unique zero h_y for $y < 0$ is greater than $2/3$, and smaller than $1/3$ for $y > 0$. Hence, given a ρ-name of $y = 0$, a continuous function producing a ρ-name of a zero of h_y would have to produce a name of a number $\geq 2/3$ and at the same time $\leq 1/3$. Impossible. □

7.5 Non-uniform version

Theorem 17 *Every computable function $f : [0,1] \to \mathbb{R}$ with $f(0) * f(1) < 0$ has a computable zero.*

Proof: We distinguish two cases:

1. the set $f^{-1}(\{0\})$ does not contain an interval. Then given a computable name of f, the algorithm of Theorem 15 will produce a computable name of a zero of f. Hence, this zero is computable.

2. the set $f^{-1}(\{0\})$ contains an interval. Then this interval contains a rational, hence a computable number.

□

Chapter 8

Complexity

We want to talk not only about computability of reals, or functions over the reals, but also about their complexity.

We restrict here to time complexity.

8.1 Measuring Complexity

The *time complexity* of a Turing machine M computing a function $f : \Sigma^\omega \to \Sigma^\omega$ should describe the asymptotic behavior of M. One would like to measure the number of time steps in dependency of the:

- output precision
- size of the input
- while keeping track of the *input lookahead*.

(dessin).

For some (good) reasons, the Cauchy representation is not suitable to define properly time complexity: basically, this is not a good idea to use the Cauchy representation, because for example for any rational number one easily construct arbitrarily long ρ names of it. This has the consequence (using a padding argument) that for any computable real x, there is a Cauchy name for x computable in linear time. This doesn't really make sense.

One uses the better *signed digit representation*: the idea is that $\overline{1}$ represents -1.

Definition 28 *The* signed digit representation $\rho_{sd} : \Sigma^\omega \to \mathbb{R}$ *is defined by:*

$$\rho_{sd}(a_n a_{n-1} \dots a_0 . a_{-1} a_{-2} \dots) := \sum_{i=n}^{-\infty} a_i . 2^i.$$

for all sequences $a_i \in \{\overline{1}, 0, 1\}$ *and* $n \geq 1$ *(with the additional properties that* $a_n \neq 0$*, if* $n \geq 0$*, and* $a_n a_{n-1} \notin \{1\overline{1}, \overline{1}1\}$ *if* $n \geq 1$*), where we interpret* $\overline{-1}$ *as* -1*.*

We will write D_i for the set of rational numbers of the form $m/2^i$ for some positive or negative integer m, and for some non-negative integer i: $D = \bigcup_{i=0}^{\infty} D_i$ is the set of *Dyadic rationals*.

In contrast to the binary representation, the signed digit representation is redundant in a symmetric way.

Definition 29 *Let* $t : \mathbb{N} \to \mathbb{N}$ *be a total function. A real number* x *is* computable in time $\mathcal{O}(t)$ *if there are constant* c *and a Turing machine that, without ever stopping, produces a* ρ_{sd}*-name of* x *and, after* $c * t(n) + c$ *steps, has written at least the prefix containing* n *digits after the binary point.*

Theorem 18 *The set of polynomial time computable real numbers forms a real algebraically closed field.*

Definition 30 *A Turing machine* M *computes a real function* $f : \mathbb{R} \to \mathbb{R}$ *on a domain* $K \subset \mathbb{R}$ *in time* $t : \mathbb{N} \to \mathbb{N}$ *with input lookahead* $l : \mathbb{N} \to \mathbb{N}$ *if and only if, for any* $p \in \rho_{sd}^{-1}(K)$*:*

1. $\rho_{sd} f_M(p) = f \rho_{sd}(p)$*: i.e.* M *computes* f *upon input* p*.*

2. *with at most* $c.t(n) + c$ *time steps (for some constant* c*).*

3. *and with reading at most* $l(n)$ *input symbols after the binary point of* p*.*

in order to produce the n*-th output symbol.*

8.2 Basic functions

Theorem 19 *For all the following functions* $f : \mathbb{R}^m \to \mathbb{R}$ *with domain* $K \subset \mathbb{R}^m$*, there exists a Turing machine* M *which computes* f *in time* t *with input lookahead* l*:*

function $f(x_1, x_2, \ldots, x_n)$	*domain* K	*time* $t(n)$	*lookahead* $l(n)$
$-x_1$	$\mathbb{R}$	n	n
$x_1 + x_2$	$[-1,1] \times [-1,1]$	n	$n + c$
$x_1 * x_2$	$[-1,1] \times [-1,1]$	$n \log n \log\log n$	$2n + c$
$x_1 * x_2$	$[-1,1] \times [-1,1]$	$n \log^2 n \log\log n$	$n + c$
$1/x_1$	$[7/8, 2]$	$n \log n \log\log n$	$2n + c$
$\exp, \sin, \cos$	$[-1,1]$	$n \log^2 n \log\log n$	$n+c$

8.3 Roots of Functions

Theorem 20 *For any computable real $x \in [0,1]$, there exists a polynomial time computable function $f : [0,1] \to \mathbb{R}$ such that f is strictly increasing, $f(0) < 0 < f(1)$, and $f^{-1}(\{0\}) = \{x\}$.*

By considering x to be computable, but not-computable in polynomial time, we get:

Corollary 5 *There exists a polynomial time computable function $f : [0,1] \to \mathbb{R}$ such that f is strictly increasing, with $f(0) < 0 < f(1)$, but $f^{-1}(\{0\})$ is not polynomial time computable.*

Proof:[of Theorem] Consider x to be computable and let $(q_n)_{n\in\mathbb{N}}$ be some sequence rapidly converging to x: $|x - q_n| < 2^{-n}$. Let $t : \mathbb{N} \to \mathbb{N}$ that bounds the run time to compute $n \mapsto q_n$.

We will define a sequence $(f_n)_{n\in\mathbb{N}}$ of simple piecewise linear functions such that $f_n \to f$, for some function f, and $f_n(x) = 0$ for all n.

The basic idea is to construct f such that for a real number y, which is close to x, and a small integer n, the machine M_f computing f at precision 2^{-n} can consider the value of f to be 0 on y. Only when the input integer n becomes so large that there is enough time to distinguish y from x, M_f will assign a non-zero value to its approximation of $f(y)$ at precision 2^{-n}, and make $f(y) \neq f(x)$.

To do so, we define two sequences $\{d_k\}$, and $\{e_k\}$ of rationals.

$$d_1 = 0,\ e_1 = 1.$$

and for $k \geq 2$,

$$d_k = \max\{d_{k-1}, q_k - 2^{-k}\} \text{ and } e_k = \min\{e_{k-1}, q_k + 2^{-k}\}.$$

We get the following facts:

- $0 = d_1 \leq d_2 \leq d_3 \leq \cdots \leq d_k < x < e_k \leq \cdots \leq e_1 = 1$.
- d_k and e_k are computable in time $O(s(k))$ where $s(k) = \sum_{i=1}^{k} t(i)$. For the sake of simplicity, we assume that $s(k) \geq k + 1$.

Then we define $\{f_n\}_n$ as follows:

- If $s(k) < n < s(k+1)$, we set $f_n = f_{s(k)}$.
- if $n = s(k)$, $k \geq 1$, we consider f_n simple piecewise linear with $\leq 2k$ break points
$$d_1, d_2, \ldots, d_k, e_k, \ldots, e_2, e_1$$
such that for all breakpoint $y \in \{d_i, e_i | 1 \leq i \leq k\}$,
$$f_n(y) = \begin{cases} -2^{-(s(i)+1)} & \text{if } y = d_i \text{ and } i = \max\{j | y = d_j\} \text{ and } i < k \\ 0 & \text{if } y = d_k \text{ or } e_k \\ 2^{-(s(i)+1)} & \text{if } y = e_i \text{ and } i = \max\{j | y = e_j\}, \text{ and } i < k \end{cases}$$

For example, if $0 = d_1 < d_2 = d_3 < d_4 < e_4 = e_3 < e_2 < e_1 = 1$, then f_{16} is piecewise linear and is determined by the following points $(d_1, -h(1))$, $(d_3, -h(3))$, $(d_4, 0)$, $(e_4, 0)$, $(e_2, h(2))$, $(e_1, h(1))$, where $h(i) = 2^{-(s(i)+1)}$.

For any $k \geq 1$, $f_{s(k)}$, $f_{s(k+1)}$ differs only on $(d_i, d_{k+1}) \cup (e_{k+1}, e_j)$ where $i = \max\{m | d_m < d_k\}$, and $j = \max\{m | e_m > e_k\}$. Thus, the maximum difference between $f_{s(k)}$ and $f_{s(k+1)}$ occurs at either d_k or e_k, and its value is at most $2^{-(s(k)+1)}$. This implies that $f = \lim_n f_n$ exists, and that

$$|f_n(y) - f(y)| \leq 2^{-n} \tag{8.1}$$

for all $n > 0$.

It is obvious that f is strictly increasing and $f(x) = 0$.

Observe that for any $k > 0$ and any i, $1 \leq i \leq k$, d_i and e_i are in D_i, and hence that if $d_i \neq d_{i+1}$ then $|d_i - d_{i+1}| > 2^{-(i+1)}$; Observe also that $|f_{s(k)}(d_i) - f_{s(k)}(d_{i+1})| \leq 2^{-s(i)}$. Thus the derivative $f'_{s(k)}$ on $[d_i, d_{i+1}]$ is bounded by $2^{-s(i)}/2^{-(i+1)} \leq 1$. Similarly, we can show this is true on the interval $[e_i, e_{i+1}]$. By mean value theorem (Théorème des accroissements finis),

$$|f_{s(k)}(y) - f_{s(k)}(d)| \leq |y - d| \tag{8.2}$$

for all $y, d \in [0, 1]$.

Consider a rational $d \in D_{s(k)}$: $f_{s(k)}(d)$ is computable in time $O(s(k))$ by computing all d_i's and e_i's $1 \leq i \leq k$, finding the pair d_i and d_j such that $d \in [d_i, d_j]$ (or finding e_i and e_j such that $d \in [e_i, e_j]$, or determining that $d \in [d_k, e_k]$), computing the values of $f(d_i)$ and $f(d_j)$ (or, respectively, computing the values of $f(e_i)$ and $f(e_j)$)) and performing a linear interpolation.

It follows that f is polynomial time computable on $[0, 1]$: indeed, to compute $f(y)$ at precision 2^{-n}, compute k such that $s(k) \geq n+1$. Take a dyadic $d \in D_{s(k)}$ at distance less than $2^{-s(k)}$ of y, then output $f_{s(k)}(d)$: this is a rational at distance less than $2^{-s(k)}$ of $f_{s(k)}(d)$ by Equation (8.2). Now, $f_{s(k)}(y)$ is at distance less than $2^{-s(k)}$ of $f(y)$ by Equation (8.1).

□

8.4 Complexity of Numerical Operators

It is difficult to define a uniform notion of complexity for operators of type $F : \mathcal{C}[0, 1] \to \mathcal{C}[0, 1]$ such as integration or differentiation.

8.4.1 Computing the maximum (optimization)

Alternatively, one can study the time complexity of $F(f)$ for polynomial-time computable f.

Theorem 21 (Friedman'84) *The following are equivalent.*

1. $P = NP$

2. *For each polynomial-time computable $f : [0,1] \to \mathbb{R}$, the maximum function $g : [0,1] \to \mathbb{R}$ defined by*

$$g(x) := \max\{f(y) | 0 \leq y \leq x\}$$

for all $x \in [0,1]$ is polynomial time computable.

Rough idea:
The direction from 1. to 2. is based on the idea that

$$z \leq g(x) \Leftrightarrow (\exists y \in [0,x]) z \leq f(y).$$

Using the Polynomial-Time Projection Theorem this is (approximately) decidable in polynomial time, if $P = NP$. By a binary search over z, one can determine $g(x)$ in polynomial time.

More formally:

Proof: $\Rightarrow$:

We only need to prove that for each $f : [0,1]^2 \to \mathbb{R}$ computable in polynomial time, the function $h : [0,1] \to \mathbb{R}$ defined by $h(x) := \max\{f(x,y) | 0 \leq y \leq 1\}$ is computable in polynomial time.

Indeed, from f computable in polynomial time over $[0,1]$, one can define function f_1 on $[0,1]^2$ computable in polynomial time as follows:

$$f_1(x,y) = \begin{cases} f(0) & \text{if } y > x \\ f(x-y) & \text{if } y \leq x \end{cases}$$

For each $x \in [0,1]$, $\max\{f_1(x,y) | 0 \leq y \leq 1\} = \max\{f(y) | 0 \leq y \leq x\}$.

Assume w.l.o.g. that $range(f) \subset [0,1]$. Consider M the Turing machine that computes f in time $p(n)$ for some polynomial p.

Define A to be the set of all pairs (d_1, e) with $e \in D_n \cap [0,1]$, $d_1 \in D_{p(n+2)} \cap [0,1]$ for some $n \geq 0$ and satisfying

$$(\exists d_2 \in D_{p(n+2)} \cap [0,1])[e \leq M^{b_{d_1}, b_{d_2}}(n+2)] :$$

here $M^{b_{d_1}, b_{d_2}}(n+2)$ denotes the digit number $n+2$ output by the machine with its two input tapes containing d_1 and d_2 respectively.

Then, $A \in NP$.

Let $d_1 \in D_{p(n+2)} \cap [0,1]$ and $e \in D_n \cap [0,1]$. Assume that for some $x \in [0,1]$, $|d_1 - x| \leq 2^{-p(n+2)}$, and $e = \max\{e_1 \in D_n \cap [0,1] | (d_1, e_1) \in A\}$. Then we claim that

1. $|e - h(d_1)| \leq 2^{-(n+1)}$
2. $|h(d_1) - h(x)| \leq 2^{-(n+2)}$

The above two claims imply that $|e - h(x)| \leq 2^{-n}$. Since e can be found by a binary search using A as oracle, it follows that function h is polynomial time computable if $P = NP$.

It remains to prove the two claims.

Proof of 1. It is immediate that $e \leq f(d_1, d_2) + 2^{-(n+2)}$ for some $d_2 \in D_{p(n+2)} \cap [0,1]$. This implies that $e - 2^{-(n+1)} \leq h(d_1)$. Conversely, assume that $h(d_1) = f(d_1, y_1)$ for some $y_1 \in [0,1]$. Choose $d_2 \in D_{p(n+2)} \cap [0,1]$ such that $|d_2 - y_1| \leq 2^{-p(n+2)}$. Then $e \geq f(d_1, d_2) - 2^{-p(n+2)} \geq f(d_1, y_1) - 2^{-(n+1)} = h(d_1) - 2^{-(n+1)}$.

Proof of 2.. Assume that $h(d_1) = f(d_1, y_1)$ and $h(x) = f(x, y_x)$. Then $|d_1 - x| \leq 2^{-p(n+2)}$ implies that $|f(d_1, y_x) - f(x, y_x)| \leq 2^{-(n+2)}$. Thus, $h(d_1) \geq f(d_1, y_x) \geq f(x, y_x) - 2^{-(n+2)} = h(x) - 2^{-(n+2)}$. Similarly, we get $h(x) \geq h(d_1) - 2^{-(n+2)}$, and thus the claim is proved.

$\Leftarrow$. Let $A \in NP$. We need to construct a function f polynomial time computable such that k polynomial time computable implies $A \in P$, where k is the maximization function such that $k(x) = \max\{f(y) | 0 \leq y \leq x\}$. Since $A \in NP$, there exists a polynomial-time predicate R and polynomial function p such that for all strings s, $s \in A \Leftrightarrow \exists t, |t| = p(|s|)\ R(s,t)$, where $|s|$ is the length of word s.

Now, we divide the interval $[0,1]$ into an infinite number of subintervals, each corresponding to a string $s \in \{0,1\}^*$. Namely, for each $n \geq 1$, let $a_n = 1 - 2^{-(n-1)}$, and for each string s of length n, if s is the ith string in $\{0,1\}^n$, $0 \leq i \leq 2^n - 1$ (i.e., s is th n-bit binary representation of integer i), then let $u_s = a_n + i * 2^{-2n}$ and $v_s = u_s + 2^{-2n}$. We further divide interval $[u_s, (u_s + v_s)/2]$ into $2^{p(n)}$ many subintervals, each corresponding to a string t of length $p(n)$. More precisely, if t is the ith string in $\{0,1\}^{p(n)}$, $0 \leq i \leq 2^{p(n)} - 1$, then we let $y_{s,t} = u_s + i * 2^{-(p(n)+2n+1)}$ and $z_{s,t} = y_{s,t} + 2^{-(p(n)+2n+1)}$.

Let g_1 be a function with $g_1(0) = 0$, $g_1(1) = 1$, g_1 strictly increasing on $[0,1]$, g_1 polynomial time computable. We can even assume $g_1^{(n)}(0) = g_1^{(n)}(1) = 0$ for all $n \geq 1$, where $g_1^{(n)}$ denotes nth derivative. We can assume also that $g_1^{(n)}$ is polynomial time computable for all n: indeed, considering

$$h(x) = \begin{cases} e^{-1/x^2} & \text{if } x > 0 \\ 0 & \text{if } x \leq 0 \end{cases}$$

then one can consider

$$g_1(x) = \frac{h(x - 1/4)}{h(3/4 - x) + h(x - 1/4)}.$$

Define a bump function $h : [0,1] \to \mathbb{R}$ by

$$h_(x) = \begin{cases} g_1(2x) & \text{if } 0 \leq x \leq 1/2 \\ g_1(2 - 2x) & \text{if } 1/2 \leq x \leq 2 \end{cases}$$

Now, we define a function f on $[0,1]$ as follows: On each interval $[(u_s + v_s)/2, v_s]$, $f(x) = 2x - v_s$. On each interval $[y_{s,t}, z_{s,t}]$,

$$f(x) = \begin{cases} u_s & \text{if not } R(s,t) \\ u_s + 2^{-(p(n)+2n+2)} * h_1(2^{p(n)+2n+1}(x - y_{s,t})) & \text{if } R(s,t) \end{cases}$$

That is f has a bump of height $2^{-(p(n)+2n+2)}$ on $[y_{s,t}, z_{s,t}]$ if t is a witness fo s being in set A, and f is flat on $[y_{s,t}, z_{s,t}]$ otherwise.

It is easy to see that f is continuous, and even infinitely derivable. Furthermore, if k is polynomial time computable over $[0,1]$, then we can determine, for each string s, whether $s \in A$ as follows: we compute an approximate value e to $k((u_s + v_s)/2)$ correct within error $\leq 2^{-(p(n)+2n+4)}$, and decide $s \in A$ iff $e > u_s$. □

8.4.2 Integration

Theorem 22 (Friedman Ko'86) *The following are equivalent.*

1. $FP = \#P$
2. *For each polynomial-time computable $f : [0,1] \to \mathbb{R}$, the integral function $g : [0,1] \to \mathbb{R}$ defined by*

$$g(x) := \int_0^x f(t)dt$$

for all $x \in [0,1]$ is polynomial time computable.

Here $\#P$ denotes the class of functions that counts the number of accepting computations of a non-deterministic polynomial-time Turing machine.

For the proof of 1. $\Rightarrow$ 2. one can guess a number of points (t, y) with $0 \leq t \leq x$ and then count those with $y \leq f(t)$ to get an approximation for the integral $g(x)$ (when f is positive).

8.5 Rough Classification

Rough classification [4]

Numerical Problem	Complexity Class
Optimisation (numbers)	NP-complete
Optimisation (functions)	EXPSPACE-complete
Ordinary Differential Equations	PSPACE-complete
Integration	#P-complete
Roots (dim 1)	P-complete
Differentiation	P (if computable)

Chapter 9

Selected Results / Myhill's Theorem / Pour-El Richard's Construction

9.1 Some Selected Theorems

9.1.1 Brouwer's Fixed Point Theorems

Theorem 23 *Every continuous function $f : [0,1]^n \to [0,1]^n$ admits a fixed point x, i.e. a point with $f(x) = x$.*

Theorem 24 (Orevkov'63) *There exists a computable function $f : [0,1]^2 \to [0,1]^2$ without computable fixed point.*

9.1.2 Derivatives

Theorem 25 (Myhill'71) *There exists a computable function $f : [0,1] \to \mathbb{R}$ with a continuous derivative $f' : [0,1] \to \mathbb{R}$ which is not computable.*

9.1.3 Differential Equations

Theorem 26 (Pour-El Richards 79) *There exists some computable*

$$f : [0,1] \times [-1,1] \to \mathbb{R}$$

such that ordinary differential equation

$$y' = f(t,y),$$

has no computable solution over any closed domain.

Theorem 27 (Pour-El Richards) *There exists a computable initial condition f and a three-dimensional wave u such that $x \mapsto u(0,x)$ is a computable function, but the unique equation of the wave equation*

$$\begin{cases} \frac{\delta^2 u}{\delta t^2} = \Delta u \\ u(0,x) = f, \frac{\delta u}{\delta t} = 0, t \in \mathbb{R}, x \in \mathbb{R}^3 \end{cases}$$

at time 1 leads to a non-computable $x \mapsto u(1,x)$.

9.2 Proof of Myhill's Theorem about Derivatives

9.2.1 General Idea

The idea is

- to construct a function f by placing "bumps" at each number of the form 2^{-n}, where $n \in \mathbb{N}$ belongs to a recursively enumerable, non-recursive set A, and by leaving the neighborhood of all other numbers 2^{-n} flat.
- for $n \in A$, the slope of the graph at 2^{-n} can be effectively bounded from below, given n: thus, if we could compute $f'(2^{-n})$ recursively, we could decide whether $n \in A$, contradicting the non-recursiveness of A.

9.2.2 A Basic Bump

We consider

$$\theta(x) = \begin{cases} x(x^2-1)^2 & \text{for } -1 \leq x \leq 1 \\ 0 & \text{for } |x| > 1 \end{cases}$$

(dessin).
Properties:

- $\theta(-1) = \theta(0) = \theta(1) = 0$
- $\theta'(-1) = \theta'(1) = 0$
- $\theta'(0) = 1$
- θ has its minimum $-\lambda$ at $x = -1/\sqrt{5}$ and its maximum $+\lambda$ at $x = 1/\sqrt{5}$.
- $|\theta(x)| \leq \lambda$ for all x.
- θ is continuous with a continuous derivative over all $\mathbb{R}$.

We call it a "bump" of length 2 and height λ.

9.2.3 A Dilated Bump

We consider $\theta_{\alpha\beta}(x) = (\beta/\lambda)\theta(x/\alpha)$: this is a bump of length 2α and height β.

(dessin)

Properties:

- $\theta_{\alpha\beta}(-\alpha) = \theta_{\alpha\beta}(0) = \theta_{\alpha\beta}(\alpha) = 0$
- $\theta'_{\alpha\beta}(-\alpha) = \theta'_{\alpha\beta}(\alpha) = 0$
- $\theta'_{\alpha\beta}(0) = \beta/(\lambda\alpha)$
- $|\theta_{\alpha\beta}(x)| \leq \beta$ for all x.
- $\theta_{\alpha\beta}$ is continuous with a continuous derivative over all $\mathbb{R}$.

9.2.4 The function f

We consider $A \subset \mathbb{N}$ recursively enumerable but non-recursive.

We consider a total recursive $h : \mathbb{N} \to \mathbb{N}$ enumerating A without repetitions: $A = \{n = h(k) | k \in \mathbb{N}\}$.

The idea is to put, for every n of the form $n = h(k)$, a bump $\theta_{\alpha_n\beta_n}$ around 2^{-n}, where $\alpha_n = 2^{-k-2n-2}$, $\beta_n = 2^{-k-n-2}$.

In other words:

$$f(x) = \sum_{k=0}^{\infty} \theta_{\alpha_{h(k)},\beta_{h(k)}}(x - 2^{-h(k)}).$$

ATTENTION: here f is written as a series, but this is not "really" a series:

- for a given x, at most one term of the sum is not zero: i.e., no two bumps overlap.
  - Indeed, the half-lengths of the bumps around the points 2^{-n} and $2^{-(n-1)}$ are at most 2^{-2n-2} and 2^{-2n}, and the sum of the two half-lengths is less than 2^{-n}, the distance between the two points.

9.2.5 The function f has a continuous derivative

f has a continuous derivative:

- the graph of f consists of alternate bumps and horizontal line-segments, and since the bumps have horizontal half-tangent at their end points, f has a continuous derivative except possibly at $x = 0$
- Given x, if x is not on a bump, then $f(x) = 0$ on a neighborhood of x, and hence $f'(x) = 0$. If x is on the bump surrounding 2^{-n}, since $|\theta'| \leq 1$ on $[0, 1]$, $|f'(x)| \leq \beta_n/\alpha_n$. Since $\beta_n/\alpha_n \to 0$ as $n \to \infty$, $f'(x) \to 0$ as $x \to 0$. From a well-known result of analysis (theorème des accroissements finis), considering that $f(0) = 0$, we have $(f(x) - f(0))/(x - 0) = f(x)/x = f'(y)$

for some $0 < y < x$. We deduce that $f(x)/x \to 0$ when $x \to 0$, that is to say that f is derivable in 0 with $f'(0) = 0$, and furthermore that f' is continuous at $x = 0$.

9.2.6 The function f is computable on $[0, 1]$

Let $x \in [0, 1]$ be given, and let it be required to compute $f(x)$ within 2^{-n}.

Let

$$g_M(x) = \sum_{k=0}^{M} \theta_{\alpha_{h(k)},\beta_{h(k)}}(x - 2^{-h(k)}).$$

it suffices to prove that

1. $|g_M(x) - f(x)| < 2^{-M}$;
2. and that $g_M(x)$ is computable.

Point 1.: for all x, $g_n(x) - f(x)$ is zero or consists of a single term

$$\theta_{\alpha_{h(k)},\beta_{h(k)}}(x - 2^{-h(k)}) = \frac{\beta_{h(k)}}{\lambda}\theta\left(\frac{x - 2^{-h(k)}}{\alpha_{h(k)}}\right).$$

But $|\theta_{\alpha_{h(k)},\beta_{h(k)}}| \leq \beta_{h(k)} = 2^{-k-h(k)-2} < 2^{-k} < 2^{-M}$ q.e.d.
Point 2.: $g_M(x)$ is computable since θ is.

- Indeed, to compute $\theta(x)$ we first determine whether $x < 0$ or $x > -1$, if $x < 0$ we can compute $\theta(x)$ since $\theta(x) = \min(0, x(x^2 - 1)^2)$. If $x > -1$, determine whether $x > 0$ or $x < 1$. If $x > 0$, then $\theta(x) = \max(0, x(x^2 - 1)^2)$. If finally $-1 < x < 1$, then $\theta(x) = x(x^2 - 1)^2$.

9.2.7 The function f' is not computable on $[0, 1]$

f' cannot be computable:

1. if $n \in A$, then $f'(2^{-n}) = \theta'_{\alpha_n\beta_n}(0) = \beta_n/(\lambda\alpha_n) = 2^{-n}/\lambda$.
2. if $n \notin A$, then $f'(2^{-n}) = 0$.

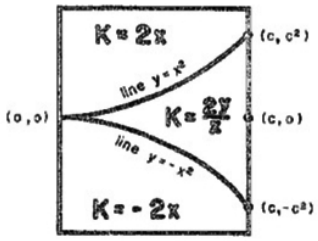
Since these alternatives can be decided, this yields a decision procedure for A.

As A is supposed to be recursively enumerable non-recursive, this is impossible.

9.3 Idea of the Proof of Pour-El Richards Theorem about Ordinary Differential Equations

9.3.1 A disperser

Consider $K(x, y)$ given by



The solutions of

$$\begin{cases} y' &= K(x,y) \\ y(0) &= 0 \end{cases}$$

are

$$y(x) = Cx^2$$

with $-1 \leq C \leq 1$.

9.3.2 A collector

We "reverse" the time variable to get a "collector".

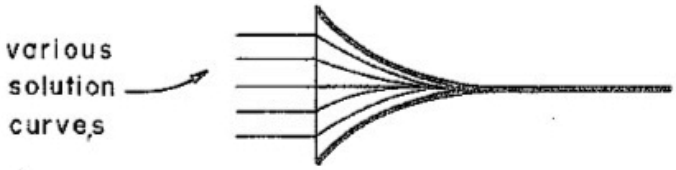


Fig. 1b. A typical collector.

9.3.3 A box

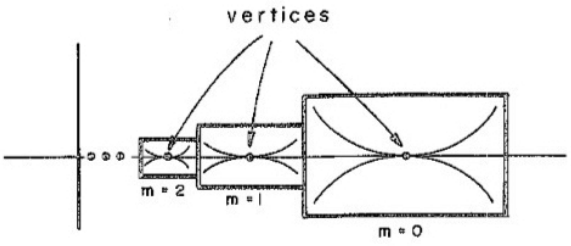
A box is made of a "collector" and of a "disperser".

(dessin).

We get differential equation $y' = F(t, y(t))$.

The trick is to consider $y' = F(t, y(t)) + h(t, y(t))$, where h is a "pulse": something null, positive, or negative. If it is positive, the solution will go above the line $y = x^2$ in the disperser, If it is negative, the solution will go below the line $y = -x^2$ in the disperser. If it is null, the solution can follow any trajectory between the parabola of the disperser.

9.3.4 A sequence of boxes



A sequence of "boxes".

- Each box is made of a "collector" and of a "disperser".
- The boxes become progressively smaller as m increases, and the vertexes converge to the origin.

This provides computability of the function.
A small "pulse" is placed at the vertex of some of the box:

- For the mth box, this pulse is positive, negative, or zero, depending on whether $m \in A$, $m \in B$, or $m \notin A \cup B$,
- where (A, B) is a fixed recursively inseparable pair of sets.

9.3.5 Idea of the proof

- Let $a(n)$ and $b(n)$ be one to one recursive functions generating A and B.
- If $a(n) = m$, then we place a positive pulse of height $2^{-(m+n+5)}$ at the vertex of box m.
- If $b(n) = m$, then we place a negative pulse of height $2^{-(m+n+5)}$ at the vertex of box m.
- Assume the solution of the ODE is computable.
- By reading $x = x_m$ at the aperture of disperser m within an error less than half the size of the aperture, one knows whether $x \in A$ or $x \in B$.
- Impossible as (A, B) is a recursively inseparable pair of sets.

Chapter 10

Bibliography

10.1 References

This document is based on (often just copied from) [**?**] and on [2].
See also book [9].
We also used [7] and [6] for Chapter 9.
We used [4] and [5] for Chapter 8.

Bibliography

[1] Vasco Brattka. Computability & complexity in analysis. Tutorial donné à *Computability in Europe (CIE'2005)*, 2005.

[2] Vasco Brattka, Peter Hertling, and Klaus Weihrauch. *New Computational Paradigms. Changing Conceptions of What is Computable*, chapter A tutorial on computable analysis. Springer-Verlag, New York, 2008.

[3] Vasco Brattka, Peter Hertling, and Klaus Weihrauch. A tutorial on computable analysis. In *New computational paradigms*, pages 425–491. Springer, 2008.

[4] Ker-I Ko. *Complexity Theory of Real Functions*. Progress in Theoretical Computer Science. Birkhaüser, Boston, 1991.

[5] Ker-I. Ko and Harvey Friedman. Computational complexity of real functions. *Theoretical Computer Science*, 20(3):323 – 352, 1982.

[6] John Myhill. A recursive function, defined on a compact interval and having a continuous derivative that is not recursive. 18:97–98, 1971.

[7] M. B. Pour-El and J. I. Richards. A computable ordinary differential equation which possesses no computable solution. *Ann. Math. Logic*, 17:61–90, 1979.

[8] Alan Turing. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society*, 42(2):230–265, 1936. Reprinted in Martin Davis. The Undecidable: Basic Papers on Undecidable Propositions, Unsolvable Problems and Computable Functions. Raven Press, 1965.

[9] K. Weihrauch. *Computable Analysis: an Introduction*. Springer, 2000.