The Inversion Problem for Computable Linear Operators

Vasco Brattka*

Theoretische Informatik I, Informatikzentrum
FernUniversität, 58084 Hagen, Germany
vasco.brattka@fernuni-hagen.de

Abstract. Given a program of a linear bounded and bijective operator T, does there exist a program for the inverse operator T^{-1}? And if this is the case, does there exist a general algorithm to transfer a program of T into a program of T^{-1}? This is the inversion problem for computable linear operators on Banach spaces in its non-uniform and uniform formulation, respectively. We study this problem from the point of view of computable analysis which is the Turing machine based theory of computability on Euclidean space and other topological spaces. Using a computable version of Banach's Inverse Mapping Theorem we can answer the first question positively. Hence, the non-uniform version of the inversion problem is solvable, while a topological argument shows that the uniform version is not. Thus, we are in the striking situation that any computable linear operator has a computable inverse while there exists no general algorithmic procedure to transfer a program of the operator into a program of its inverse. As a consequence, the computable version of Banach's Inverse Mapping Theorem is a powerful tool which can be used to produce highly non-constructive existence proofs of algorithms. We apply this method to prove that a certain initial value problem admits a computable solution.

Keywords: computable analysis, linear operators, inversion problem.

1 Introduction

Given two Banach spaces X, Y and a linear bounded and bijective operator $T : X \to Y$, Banach's Inverse Mapping Theorem [10] guarantees that the inverse $T^{-1} : Y \to X$ is a linear bounded operator as well. Hence, it is reasonable to ask for computable versions of this fact, i.e. do the implications

(1) T computable $\Longrightarrow$ T^{-1} computable **(non-uniform inversion problem)**,
(2) $T \mapsto T^{-1}$ computable **(uniform inversion problem)**

hold? Of course, both questions have to be specified carefully. In particular, the computability notion in the uniform case should reflect the fact that algorithms of T are transfered into algorithms of T^{-1}.

* Work partially supported by DFG Grant BR 1807/4-1

Such meaningful computability notions are provided by computable analysis, which is the Turing machine based theory of computability on real numbers and other topological spaces. Pioneering work on this theory has been presented by Turing [22], Banach and Mazur [1], Lacombe [17] and Grzegorczyk [11]. Recent monographs have been published by Pour-El and Richards [19], Ko [15] and Weihrauch [24]. Certain aspects of computable functional analysis have already been studied by several authors, see for instance [18, 9, 23, 27, 28, 25, 26].

From the computational point of view Banach's Inverse Mapping Theorem is interesting, since its classical proof relies on the Baire Category Theorem and therefore it counts as "non-constructive" (see [5] for a discussion of computable versions of the Baire Category Theorem). However, a "non-constructive" application of the Baire Category Theorem suffices in order to prove that the non-uniform inversion problem (1) is solvable; but at the same time the non-constructiveness is the reason why the uniform inversion problem (2) is not solvable.

We close the introduction with a short survey of the organisation of this paper. In the following Section 2 we will present some preliminaries from computable analysis. In Section 3 we discuss computable metric spaces, computable Banach spaces and effective open subsets. In Section 4 we investigate computable versions of the Open Mapping Theorem and based on these results we study Banach's Inverse Mapping Theorem in Section 5. Finally, in Section 6 we apply the computable version of this theorem in order to prove that a certain initial value problem admits a computable solution. In this extended abstract most proofs are omitted; they can be found in [4] (or in the Appendix).

2 Preliminaries from Computable Analysis

In this section we briefly summarize some notions from computable analysis. For details the reader is refered to [24]. The basic idea of the representation based approach to computable analysis is to represent infinite objects like real numbers, functions or sets, by infinite strings over some alphabet Σ (which should at least contain the symbols 0 and 1). Thus, a *representation* of a set X is a surjective mapping $\delta :\subseteq \Sigma^\omega \to X$ and in this situation we will call (X, δ) a *represented space*. Here Σ^ω denotes the set of infinite sequences over Σ and the inclusion symbol is used to indicate that the mapping might be partial. If we have two represented spaces, then we can define the notion of a computable function.

Definition 1 (Computable function). Let (X, δ) and (Y, δ') be represented spaces. A function $f :\subseteq X \to Y$ is called (δ, δ')*–computable*, if there exit some computable function $F :\subseteq \Sigma^\omega \to \Sigma^\omega$ such that $\delta' F(p) = f\delta(p)$ for all $p \in \mathrm{dom}(f\delta)$.

Of course, we have to define computability of functions $F :\subseteq \Sigma^\omega \to \Sigma^\omega$ to make this definition complete, but this can be done via Turing machines: F is computable if there exists some Turing machine, which computes infinitely

long and transforms each sequence p, written on the input tape, into the corresponding sequence $F(p)$, written on the one-way output tape. Later on, we will also need computable multi-valued operations $f :\subseteq X \rightrightarrows Y$, which are defined analogously to computable functions by substituting $\delta' F(p) \in f\delta(p)$ for the equation in Definition 1 above. If the represented spaces are fixed or clear from the context, then we will simply call a function or operation f *computable*.

For the comparison of representations it will be useful to have the notion of *reducibility* of representations. If δ, δ' are both representations of a set X, then δ is called *reducible* to δ', $\delta \leq \delta'$ in symbols, if there exists a computable function $F :\subseteq \Sigma^\omega \to \Sigma^\omega$ such that $\delta(p) = \delta' F(p)$ for all $p \in \mathrm{dom}(\delta)$. Obviously, $\delta \leq \delta'$ holds, if and only if the identity $\mathrm{id} : X \to X$ is (δ, δ')–computable. Moreover, δ and δ' are called *equivalent*, $\delta \equiv \delta'$ in symbols, if $\delta \leq \delta'$ and $\delta' \leq \delta$.

Analogously to the notion of computability we can define the notion of (δ, δ')–*continuity* for single- and multi-valued operations, by substituting a continuous function $F :\subseteq \Sigma^\omega \to \Sigma^\omega$ for the computable function F in the definitions above. On Σ^ω we use the *Cantor topology*, which is simply the product topology of the discrete topology on Σ. The corresponding reducibility will be called *continuous reducibility* and we will use the symbols $\leq_t$ and $\equiv_t$ in this case. Again we will simply say that the corresponding function is *continuous*, if the representations are fixed or clear from the context. If not mentioned otherwise, we will always assume that a represented space is endowed with the final topology induced by its representation.

This will lead to no confusion with the ordinary topological notion of continuity, as long as we are dealing with *admissible* representations. A representation δ of a topological space X is called *admissible*, if δ is maximal among all continuous representations δ' of X, i.e. if $\delta' \leq_t \delta$ holds for all continuous representations δ' of X. If δ, δ' are admissible representations of topological spaces X, Y, then a function $f :\subseteq X \to Y$ is (δ, δ')–continuous, if and only if it is sequentially continuous, cf. [20, 6].

Given a represented space (X, δ), we will occasionally use the notions of a *computable sequence* and a *computable point*. A *computable sequence* is a computable function $f : \mathbb{N} \to X$, where we assume that $\mathbb{N} = \{0, 1, 2, ...\}$ is represented by $\delta_{\mathbb{N}}(1^n 0^\omega) := n$ and a point $x \in X$ is called *computable*, if there is a constant computable sequence with value x.

Given two represented spaces (X, δ) and (Y, δ'), there is a canonical representation $[\delta, \delta']$ of $X \times Y$ and a representation $[\delta \to \delta']$ of certain functions $f : X \to Y$. If δ, δ' are *admissible* representations of sequential topological spaces, then $[\delta \to \delta']$ is actually a representation of the set $\mathcal{C}(X, Y)$ of continuous functions $f : X \to Y$. If $Y = \mathbb{R}$, then we write for short $\mathcal{C}(X) := \mathcal{C}(X, \mathbb{R})$. The function space representation can be characterized by the fact that it admits evaluation and type conversion.

Proposition 2 (Evaluation and type conversion). *Let (X, δ) and (Y, δ') be admissibly represented sequential topological spaces and let (Z, δ'') be a represented space. Then:*

(1) **(Evaluation)** $\mathrm{ev} : \mathcal{C}(X,Y) \times X \to Y, (f,x) \mapsto f(x)$ *is* $([[\delta \to \delta'], \delta], \delta')$*–computable,*

(2) **(Type conversion)** $f : Z \times X \to Y$*, is* $([\delta'', \delta], \delta')$*–computable, if and only if the function* $\check{f} : Z \to \mathcal{C}(X,Y)$*, defined by* $\check{f}(z)(x) := f(z,x)$ *is* $(\delta'', [\delta \to \delta'])$*–computable.*

The proof of this proposition is based on a version of smn– and utm-Theorem, see [24, 20]. If (X,δ), (Y,δ') are admissibly represented sequential topological spaces, then in the following we will always assume that $\mathcal{C}(X,Y)$ is represented by $[\delta \to \delta']$. It follows by evaluation and type conversion that the computable points in $(\mathcal{C}(X,Y), [\delta \to \delta'])$ are just the (δ,δ')–computable functions $f : X \to Y$. Since evaluation and type conversion are even characteristic properties of the function space representation $[\delta \to \delta']$, it follows that this representation actually reflects the properties of programs. That is, a name p of a function $f = [\delta \to \delta'](p)$ can be considered as a "program" of f since it just contains sufficiently much information in order to evaluate f. This corresponds to the well-known fact that the compact-open topology is the appropriate topology for programs [21] and actually, if (X,δ), (Y,δ') are admissibly represented separable Banach spaces, one obtains the compact open topology as final topology of $[\delta \to \delta']$ (see [20]).

If (X,δ) is a represented space, then we will always assume that the set of sequences $X^{\mathbb{N}}$ is represented by $\delta^{\mathbb{N}} := [\delta_{\mathbb{N}} \to \delta]$. The computable points in $(X^{\mathbb{N}}, \delta^{\mathbb{N}})$ are just the computable sequences in (X,δ). Moreover, we assume that X^n is always represented by δ^n, which can be defined inductively by $\delta^1 := \delta$ and $\delta^{n+1} := [\delta^n, \delta]$.

3 Computable Metric and Banach Spaces

In this section we will briefly discuss computable metric spaces and computable Banach spaces. The notion of a computable Banach space will be the central notion for all following results. Computable metric spaces have been used in the literature at least since Lacombe [17]. Pour-El and Richards have introduced a closely related axiomatic characterization of sequential computability structures for Banach spaces [19] which has been extended to metric spaces by Mori, Tsujii, and Yasugi [27].

We mention that we will denote the *open balls* of a metric space (X,d) by $B(x,\varepsilon) := \{y \in X : d(x,y) < \varepsilon\}$ for all $x \in X$, $\varepsilon > 0$ and correspondingly *closed balls* by $\overline{B}(x,\varepsilon) := \{y \in X : d(x,y) \leq \varepsilon\}$. Occasionally, we denote complements of sets $A \subseteq X$ by $A^c := X \setminus A$.

Definition 3 (Computable metric space). A tuple (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 which is dense in X,
(3) $d \circ (\alpha \times \alpha) : \mathbb{N}^2 \to \mathbb{R}$ is a computable (double) sequence in $\mathbb{R}$.

Here, we tacitly assume that the reader is familiar with the notion of a computable sequence of reals, but we will come back to that point below. Occasionally, we will say for short that X is a *computable metric space*. Obviously, a computable metric space is especially separable. Given a computable metric space (X, d, α), its *Cauchy representation* $\delta_X :\subseteq \Sigma^\omega \to X$ can be defined by

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

for all n_i such that $(\alpha(n_i))_{i\in\mathbb{N}}$ converges and $d(\alpha(n_i), \alpha(n_j)) \leq 2^{-i}$ for all $j > i$ (and undefined for all other input sequences). In the following we tacitly assume that computable metric spaces are represented by their Cauchy representations. If X is a computable metric space, then it is easy to see that $d : X \times X \to \mathbb{R}$ becomes computable [6]. All Cauchy representations are admissible with respect to the corresponding metric topology.

An important computable metric space is $(\mathbb{R}, d_\mathbb{R}, \alpha_\mathbb{R})$ with the Euclidean metric $d_\mathbb{R}(x, y) := |x - y|$ and some standard numbering of the rational numbers $\mathbb{Q}$, as $\alpha_\mathbb{R}\langle i, j, k\rangle := (i-j)/(k+1)$. Here, $\langle i, j\rangle := 1/2(i+j)(i+j+1)+j$ denotes *Cantor pairs* and this definition is extended inductively to finite tuples. Similarly, we can define $\langle p, q\rangle \in \Sigma^\omega$ for sequences $p, q \in \Sigma^\omega$. For short we will occasionally write $\overline{k} := \alpha_\mathbb{R}(k)$. In the following we assume that $\mathbb{R}$ is endowed with the Cauchy representation $\delta_\mathbb{R}$ induced by the computable metric space given above. This representation of $\mathbb{R}$ can also be defined, if $(\mathbb{R}, d_\mathbb{R}, \alpha_\mathbb{R})$ just fulfills (1) and (2) of the definition above and this leads to a definition of computable real number sequences without circularity. Occasionally, we will also use the represented space $(\mathbb{Q}, \delta_\mathbb{Q})$ of rational numbers with $\delta_\mathbb{Q}(1^n0^\omega) := \overline{n}$. Computationally, we do not have to distinguish the complex numbers $\mathbb{C}$ from $\mathbb{R}^2$. We will use the notation $\mathbb{F}$ for a field which always might be replaced by both, $\mathbb{R}$ or $\mathbb{C}$. Correspondingly, we use the notation $(\mathbb{F}, d_\mathbb{F}, \alpha_\mathbb{F})$ for a computable metric space which might be replaced by both computable metric spaces $(\mathbb{R}, d_\mathbb{R}, \alpha_\mathbb{R})$ and $(\mathbb{C}, d_\mathbb{C}, \alpha_\mathbb{C})$ (defined analogously). We will also use the notation $Q_\mathbb{F} = \text{range}(\alpha_\mathbb{F})$, i.e. $Q_\mathbb{R} = \mathbb{Q}$ and $Q_\mathbb{C} = \mathbb{Q}[i]$.

For the definition of a computable Banach space it is helpful to have the notion of a computable vector space which we will define next.

Definition 4 (Computable vector space). A represented space (X, δ) is called a *computable vector space* (over $\mathbb{F}$), if $(X, +, \cdot, 0)$ is a vector space over $\mathbb{F}$ such that the following conditions hold:

(1) $+ : X \times X \to X, (x, y) \mapsto x + y$ is computable,
(2) $\cdot : \mathbb{F} \times X \to X, (a, x) \mapsto a \cdot x$ is computable,
(3) $0 \in X$ is a computable point.

If (X, δ) is a computable vector space over $\mathbb{F}$, then $(\mathbb{F}, \delta_\mathbb{F})$, (X^n, δ^n) and $(X^\mathbb{N}, \delta^\mathbb{N})$ are computable vector spaces over $\mathbb{F}$. If, additionally, (X, δ), (Y, δ') are admissibly represented second countable T_0–spaces, then the function space $(\mathcal{C}(Y, X), [\delta' \to \delta])$ is a computable vector space over $\mathbb{F}$. Here we tacitly assume that the vector space operations on product, sequence and function spaces are

defined componentwise. The proof for the function space is a straightforward application of evaluation and type conversion. The central definition for the present investigation will be the notion of a computable Banach space.

Definition 5 (Computable normed space). A tuple $(X, ||\ ||, e)$ is called a *computable normed space*, if

(1) $||\ || : X \to \mathbb{R}$ is a norm on X,
(2) $e : \mathbb{N} \to X$ is a *fundamental sequence*, i.e. its linear span is dense in X,
(3) (X, d, α_e) with $d(x, y) := ||x - y||$ and $\alpha_e\langle k, \langle n_0, ..., n_k\rangle\rangle := \sum_{i=0}^{k} \alpha_{\mathbb{F}}(n_i)e_i$, is a computable metric space with Cauchy representation δ_X,
(4) (X, δ_X) is a computable vector space over $\mathbb{F}$.

If in the situation of the definition the underlying space $(X, ||\ ||)$ is even a Banach space, i.e. if (X, d) is a complete metric space, then $(X, ||\ ||, e)$ is called a *computable Banach space*. If the norm and the fundamental sequence are clear from the context or locally irrelevant, we will say for short that X is a *computable normed space* or a *computable Banach space*. We will always assume that computable normed spaces are represented by their Cauchy representations, which are admissible with respect to the norm topology. If X is a computable normed space, then $||\ || : X \to \mathbb{R}$ is a computable function. Of course, all computable Banach spaces are separable. In the following proposition a number of computable Banach spaces are defined.

Proposition 6 (Computable Banach spaces). *Let $p \in \mathbb{R}$ be a computable real number with $1 \leq p < \infty$ and let $a < b$ be computable real numbers. The following spaces are computable Banach spaces over $\mathbb{F}$.*

(1) $(\mathbb{F}^n, ||\ ||_\infty, e)$ *with*
- $||(x_1, x_2, ..., x_n)||_\infty := \max\limits_{k=1,...,n} |x_k|$,
- $e_i = e(i) = (e_{i1}, e_{i2}, ..., e_{in})$ *with* $e_{ik} := \begin{cases} 1 \text{ if } i = k \\ 0 \text{ else} \end{cases}$.

(2) $(\ell_p, ||\ ||_p, e)$ *with*
- $\ell_p := \{x \in \mathbb{F}^{\mathbb{N}} : ||x||_p < \infty\}$,
- $||(x_k)_{k\in\mathbb{N}}||_p := \sqrt[p]{\sum\limits_{k=0}^{\infty} |x_k|^p}$,
- $e_i = e(i) = (e_{ik})_{k\in\mathbb{N}}$ *with* $e_{ik} := \begin{cases} 1 \text{ if } i = k \\ 0 \text{ else} \end{cases}$.

(3) $(\mathcal{C}^{(n)}[a, b], ||\ ||_{(n)}, e)$ *with*
- $\mathcal{C}^{(n)}[a, b] := \{f : [a, b] \to \mathbb{R} : f \text{ } n\text{–times continuously differentiable}\}$,
- $||f||_{(n)} := \sum\limits_{i=0}^{n} \max\limits_{t\in[a,b]} |f^{(i)}(t)|$,
- $e_i(t) = e(i)(t) = t^i$.

We leave it to the reader to check that these spaces are actually computable Banach spaces. If not stated differently, then we will assume that $(\mathbb{F}^n, ||\ ||)$ is endowed with the maximum norm $||\ ||_\infty$. It is known that the Cauchy representation $\delta_{\mathcal{C}[a,b]}$ of $\mathcal{C}^{(0)}[a, b] = \mathcal{C}([a, b], \mathbb{R})$ is equivalent to $[\delta_{[a,b]} \to \delta_{\mathbb{R}}]$, where $\delta_{[a,b]}$

denotes the restriction of $\delta_{\mathbb{R}}$ to $[a,b]$ (cf. Lemma 6.1.10 in [24]). In the following we will occasionally utilize the sequence spaces ℓ_p to construct counterexamples.

Since we will study the Open Mapping Theorem in the next section, we have to compute with open sets. Therefore we need representations of the hyperspace $\mathcal{O}(X)$ of open subsets of X. Such representations have been studied in the Euclidean case in [8, 24] and for the metric case in [7].

Definition 7 (Hyperspace of open subsets). Let (X,d,α) be a computable metric space. We endow the hyperspace $\mathcal{O}(X) := \{U \subseteq X : U \text{ open}\}$ with the representation $\delta_{\mathcal{O}(X)}$, defined by $\delta_{\mathcal{O}(X)}(p) := \bigcup_{i=0}^{\infty} B(\alpha(n_i), \overline{k_i})$ for all sequences $p = 01^{\langle n_0,k_0\rangle+1}01^{\langle n_1,k_1\rangle+1}01^{\langle n_2,k_2\rangle+1}...$ with $n_i, k_i \in \mathbb{N}$.

Those open subsets $U \subseteq X$ which are computable points in $\mathcal{O}(X)$ are called *r.e. open*. We close this section with one helpful proposition which states that we can represent open subsets by preimages of continuous functions. This is a direct consequence of results in [7] and an effective version of the statement that open subsets of metric spaces coincide with the functional open subsets.

Proposition 8 (Functional open subsets). *Let X be a computable metric space. The map $Z : \mathcal{C}(X) \to \mathcal{O}(X), f \mapsto X \setminus f^{-1}\{0\}$ is computable and admits a computable right-inverse $\mathcal{O}(X) \rightrightarrows \mathcal{C}(X)$.*

4 The Open Mapping Theorem

In this section we will study the effective content of the Open Mapping Theorem, which we formulate first. The classical proof of this theorem can be found in [10] or other textbooks on functional analysis.

Theorem 9 (Open Mapping Theorem). *Let X, Y be Banach spaces. If $T : X \to Y$ is a linear surjective and bounded operator, then T is open, i.e. $T(U) \subseteq Y$ is open for any open $U \subseteq X$.*

Whenever $T : X \to Y$ is an open operator, we can associate the function

$$\mathcal{O}(T) : \mathcal{O}(X) \to \mathcal{O}(Y), U \mapsto T(U)$$

with it. Now we can ask for three different computable versions of the Open Mapping Theorem. If $T : X \to Y$ is a linear computable and surjective operator, does the following hold true:

(1) $U \subseteq X$ r.e. open $\Longrightarrow T(U) \subseteq Y$ r.e. open?
(2) $\mathcal{O}(T) : \mathcal{O}(X) \to \mathcal{O}(Y), U \mapsto T(U)$ is computable?
(3) $T \mapsto \mathcal{O}(T)$ is computable?

Since any computable function maps computable inputs to computable outputs, we can conclude (3)$\Longrightarrow$(2)$\Longrightarrow$(1). In the following we will see that questions (1) and (2) can be answered in the affirmative, while question (3) has to be answered in the negative. The key tool for the positive results will be Theorem 10 on effective openness. It states that $T : X \to Y$ is computable, if and only if $\mathcal{O}(T) : \mathcal{O}(X) \to \mathcal{O}(Y)$ is computable, provided that T is a linear bounded and and open operator and X, Y are computable normed spaces.

Theorem 10 (Effective openness). *Let X, Y be computable normed spaces and let $T : X \to Y$ be a linear and bounded operator. Then the following conditions are equivalent:*

(1) $T : X \to Y$ is open and computable,
(2) $\mathcal{O}(T) : \mathcal{O}(X) \to \mathcal{O}(Y), U \mapsto T(U)$ is well-defined and computable.

The proof of "(1)$\Longrightarrow$(2)" can be performed in two steps. First one uses the fact that T is open and linear in order to prove that given $x \in X$ and an open subset $U \subseteq X$, we can effectively find some radius $r > 0$ such that $B(Tx, r) \subseteq T(U)$. This is the non-uniform part of the proof since it is based on the fact that for any open T there exists some radius $r' > 0$ with $B(0, r') \subseteq T(B(0, 1))$ (such an r' always exists, but it can not be effectively determined from T). In the second step, computability of T is exploited in order to prove that $\mathcal{O}(T)$ is computable. Using the previous theorem we can directly conclude a computable version of the Open Mapping Theorem as a corollary of the classical Open Mapping Theorem.

Corollary 11 (Computable Open Mapping Theorem). *Let X, Y be computable Banach spaces and let $T : X \to Y$ be a linear computable operator. If T is surjective, then T is open and $\mathcal{O}(T) : \mathcal{O}(X) \to \mathcal{O}(Y)$ is computable. Especially, $T(U) \subseteq Y$ is r.e. open for any r.e. open set $U \subseteq X$.*

This version of the Open Mapping Theorem leaves open the question whether the map $T \mapsto \mathcal{O}(T)$ itself is computable. This question is answered negatively by the following example.

Example 12. The mapping $T \mapsto \mathcal{O}(T)$, defined for linear, bounded and bijective operators $T : \ell_2 \to \ell_2$ with $||T|| = 1$, is not $([\delta_{\ell_2} \to \delta_{\ell_2}], \delta_{\mathcal{O}(\ell_2)})$–continuous.

We leave the proof to the reader (see [4]). Although the mapping $T \mapsto \mathcal{O}(T)$ is discontinuous, we know by Theorem 10 that $\mathcal{O}(T) : \mathcal{O}(\ell_2) \to \mathcal{O}(\ell_2)$ is computable whenever $T : \ell_2 \to \ell_2$ is computable. On the one hand, this guarantees that $T \mapsto \mathcal{O}(T)$ is not too discontinuous [3]. On the other hand, we have to use sequences to construct a computable counterexample for the uniform version of the Open Mapping Theorem. The proof is based on appropriately defined diagonal matrices.

Proposition 13. *There exists a computable sequence $(T_n)_{n\in\mathbb{N}}$ in $\mathcal{C}(\ell_2, \ell_2)$ of linear computable and bijective operators $T_n : \ell_2 \to \ell_2$ such that $(T_n B(0, 1))_{n\in\mathbb{N}}$ is a sequence of r.e. open subsets of ℓ_2 which is not computable in $\mathcal{O}(\ell_2)$.*

5 Banach's Inverse Mapping Theorem

In this section we want to study computable versions of Banach's Inverse Mapping Theorem. Again we start with a formulation of the classical theorem.

Theorem 14 (Banach's Inverse Mapping Theorem). *Let X, Y be Banach spaces and let $T : X \to Y$ be a linear bounded operator. If T is bijective, then $T^{-1} : Y \to X$ is bounded.*

Similarly as in case of the Open Mapping Theorem we have two canonical candidates for an effective version of this theorem: the non-uniform version (1) and the uniform version (2), as formulated in the Introduction. Again we will see that the non-uniform version admits a solution while the uniform version does not. Analogously as we have used Theorem 10 on effective openness to prove the computable Open Mapping Theorem 11, we will use Theorem 15 on effective continuity to prove the computable version of Banach's Inverse Mapping Theorem. We sketch the first part of the proof which is based on Proposition 8.

Theorem 15 (Effective continuity). *Let X, Y be computable metric spaces and let $T : X \to Y$ be a function. Then the following conditions are equivalent:*

(1) $T : X \to Y$ is computable,
(2) $\mathcal{O}(T^{-1}) : \mathcal{O}(Y) \to \mathcal{O}(X), V \mapsto T^{-1}(V)$ is well-defined and computable.

Proof. "(1)$\Longrightarrow$(2)" If $T : X \to Y$ is computable, then it is continuous and hence $\mathcal{O}(T^{-1})$ is well-defined. Given a function $f : Y \to \mathbb{R}$ such that $V = Y \setminus f^{-1}\{0\}$, we obtain $T^{-1}(V) = T^{-1}(Y \setminus f^{-1}\{0\}) = X \setminus (fT)^{-1}\{0\}$. Using Proposition 8 and the fact that composition $\circ : \mathcal{C}(Y, \mathbb{R}) \times \mathcal{C}(X, Y) \to \mathcal{C}(X, \mathbb{R}), (f, T) \mapsto f \circ T$ is computable (which can be proved by evaluation and type conversion), we obtain that $\mathcal{O}(T^{-1})$ is computable. □

Now we note the fact that for Banach spaces X, Y and bijective linear operators $T : X \to Y$, the operation $\mathcal{O}(T^{-1})$, associated with T according to the previous theorem, is the same as the operation $\mathcal{O}(S)$, associated with $S = T^{-1}$ according to Theorem 10. Thus, we can directly conclude the following computable version of the Inverse Mapping Theorem as a corollary of Theorem 15 and Theorem 10.

Corollary 16 (Computable Inverse Mapping Theorem). *Let X, Y be computable Banach spaces and let $T : X \to Y$ be a linear computable operator. If T is bijective, then $T^{-1} : Y \to X$ is computable too.*

In contrast to the Theorem 10 on effective openness, we can formulate a uniform version of the Theorem 15 on effective continuity: $\omega : T \mapsto \mathcal{O}(T^{-1})$, defined for all continuous $T : X \to Y$, is $([\delta_X \to \delta_Y], [\delta_{\mathcal{O}(Y)} \to \delta_{\mathcal{O}(X)}])$–computable and its inverse ω^{-1} is computable in the corresponding sense too. Using this positive result, applied to T^{-1}, we can transfer our negative results on the Open Mapping Theorem to the Inverse Mapping Theorem. As a corollary of this fact and Example 12 we obtain the following result.

Example 17. The inversion map $T \mapsto T^{-1}$, defined for linear bounded and bijective operators $T : \ell_2 \to \ell_2$ with $||T|| = 1$, is not continuous.

This holds with respect to $([\delta_{\ell_2} \to \delta_{\ell_2}], [\delta_{\ell_2} \to \delta_{\ell_2}])$–continuity, that is with respect to the compact open topology. Correspondingly, we can construct a computable counterexample for the uniform version of the Inverse Mapping Theorem. As a corollary of Proposition 13 we obtain the following counterexample.

Corollary 18. *There exists a computable sequence $(T_n)_{n\in\mathbb{N}}$ in $\mathcal{C}(\ell_2,\ell_2)$ of linear computable and bijective operators $T_n:\ell_2\to\ell_2$, such that $(T_n^{-1})_{n\in\mathbb{N}}$ is a sequence of computable operators $T_n^{-1}:\ell_2\to\ell_2$ which is not computable in $\mathcal{C}(\ell_2,\ell_2)$.*

We can even assume that $||T_n||=1$ for all $n\in\mathbb{N}$. We close this section with an application of the Computable Inverse Mapping Theorem 16 which shows that any two comparable computable complete norms are computably equivalent.

Theorem 19. *Let $(X,||\ ||)$, $(X,||\ ||')$ be computable Banach spaces and let δ,δ' be the corresponding Cauchy representations of X. If $\delta\leq\delta'$ then $\delta\equiv\delta'$.*

Proof. If $\delta\leq\delta'$, then the identity $\mathrm{id}:(X,||\ ||)\to(X,||\ ||')$ is (δ,δ')–computable. Moreover, the identity is obviously linear and bijective. Thus, the inverse identity $\mathrm{id}^{-1}:(X,||\ ||')\to(X,||\ ||)$ is (δ',δ)–computable by the Computable Inverse Mapping Theorem. Consequently, $\delta'\leq\delta$. □

6 An Initial Value Problem

In this section we will discuss an application of the computable version of Banach's Inverse Mapping Theorem to the initial value problem of ordinary linear differential equations. Consider the linear differential equation with initial values

$$\sum_{i=0}^{n} f_i(t)x^{(i)}(t)=y(t) \text{ with } x^{(j)}(0)=a_j \text{ for } j=0,...,n-1. \tag{1}$$

Here, $x,y:[0,1]\to\mathbb{R}$ are functions, $f_i:[0,1]\to\mathbb{R}$ are coefficient functions with $f_n\neq 0$ and $a_0,...,a_{n-1}\in\mathbb{R}$ are initial values. It is known that for each $y\in\mathcal{C}[0,1]$ and all values $a_0,...,a_{n-1}$ there is exactly one solution $x\in\mathcal{C}^{(n)}[0,1]$ of this equation [12]. Given f_i,a_i and y, can we effectively find this solution? The positive answer to this question can easily be deduced from the computable Inverse Mapping Theorem 16.

Theorem 20 (Initial Value Problem). *Let $n\geq 1$ be a natural number and let $f_0,...,f_n:[0,1]\to\mathbb{R}$ be computable functions with $f_n\neq 0$. The solution operator $L:\mathcal{C}[0,1]\times\mathbb{R}^n\to\mathcal{C}^{(n)}[0,1]$ which maps each tuple $(y,a_0,...,a_{n-1})\in\mathcal{C}[0,1]\times\mathbb{R}^n$ to the unique function $x=L(y,a_0,...,a_{n-1})$ which fulfills Equation (1), is computable.*

Proof. $L^{-1}:\mathcal{C}^{(n)}[0,1]\to\mathcal{C}[0,1]\times\mathbb{R}^n, x\mapsto\left(\sum_{i=0}^n f_ix^{(i)},x^{(0)}(0),...,x^{(n-1)}(0)\right)$ is obviously linear. Using the evaluation and type conversion property and the fact that the i–th differentiation operator $\mathcal{C}^{(n)}[0,1]\to\mathcal{C}[0,1], x\mapsto x^{(i)}$ is computable for $i\leq n$, one can easily prove that L^{-1} is computable. By the computable Inverse Mapping Theorem 16 it follows that L is computable too. □

We obtain the following immediate corollary on computability of solutions of ordinary linear differential equations.

Corollary 21. *Let $n\geq 1$ and let $y,f_0,...,f_n:[0,1]\to\mathbb{R}$ be computable functions and let $a_0,...,a_{n-1}\in\mathbb{R}$ be computable real numbers. Then the unique function $x\in\mathcal{C}^{(n)}[0,1]$ which fulfills Equation (1) is a computable point in $\mathcal{C}^{(n)}[0,1]$. Especially, $x^{(0)},...,x^{(n)}:[0,1]\to\mathbb{R}$ are computable functions.*

7 Conclusion

We have investigated the non-uniform and the uniform version of the inversion problem for computable linear operators. The computable version of Banach's Inverse Mapping Theorem, Corollary 16, shows that the non-uniform version is solvable while Example 17 proves by a topological argument and Corollary 18 proves by a computability argument that the uniform version is not solvable (this is because any computable operation is continuous and maps computable sequences to computable sequences, respectively).

The negative result corresponds to what is known in constructive analysis [2]. However, our positive results on Banach's Inverse Mapping Theorem cannot be deduced from known positive results in constructive analysis [13, 14] since these theorems have stronger assumptions (such as "effective bijectivity").

In a certain sense, the negative result seems to be in contrast with the so-called Banach's Inversion Stability Theorem [16]. However, this theorem states that $T \mapsto T^{-1}$ is continuous with respect to the *operator norm topology* on the function space. In the infinite-dimensional case this topology is different from the compact open topology and it is only the latter which reflects the meaning of "programs". Additionally, the operator norm topology is not separable in these cases and hence it is not obvious how to handle it computationally [4]. Moreover, it is worth mentioning that in case of finite-dimensional spaces X, Y, the uniform version of the inversion problem becomes solvable as well [4]. In this case $T \mapsto T^{-1}$ is computable (and continuous with respect to the compact open topology).

Altogether we are in the somewhat surprising situation that in the general case of infinite-dimensional Banach spaces X, Y any computable linear operator $T : X \to Y$ has a computable inverse T^{-1} while there is no general algorithmic procedure to transfer programs of T into programs of T^{-1}. As we have demonstrated with Theorem 20, this leads to highly non-effective existence proofs of algorithms: the proof of Theorem 20 shows that there exist an algorithm which solves the corresponding initial value problem without a single hint how such an algorithm could look like. Nevertheless, this is a meaningful insight, since only in case of existence the search for a concrete algorithm is promising (of course, in the special case of the initial value problem such concrete algorithms are known).

References

1. S. Banach and S. Mazur, Sur les fonctions calculables, *Ann. Soc. Pol. de Math.* **16** (1937) 223.
2. E. Bishop and D. S. Bridges, *Constructive Analysis*, Springer, Berlin 1985.
3. V. Brattka, Computable invariance, *Theoret. Comp. Sci.* **210** (1999) 3–20.
4. V. Brattka, Computability of Banach space principles, Informatik Berichte 286, FernUniversität Hagen, Fachbereich Informatik, Hagen, June 2001.
5. V. Brattka, Computable versions of Baire's category theorem, in: J. Sgall, A. Pultr, and P. Kolman (eds.), *Mathematical Foundations of Computer Science 2001*, vol. 2136 of *Lect. Not. Comp. Sci.*, Springer, Berlin 2001, 224–235.

6. V. Brattka, Computability over topological structures, in: S. B. Cooper and S. Goncharov (eds.), *Computability and Models*, Kluwer Academic Publishers, Dordrecht (in preparation).
7. V. Brattka and G. Presser, Computability on subsets of metric spaces, *Theoret. Comp. Sci.* (accepted for publication).
8. V. Brattka and K. Weihrauch, Computability on subsets of Euclidean space I: Closed and compact subsets, *Theoret. Comp. Sci.* **219** (1999) 65–93.
9. X. Ge and A. Nerode, Effective content of the calculus of variations I: semicontinuity and the chattering lemma, *Ann. Pure Appl. Logic* **78** (1996) 127–146.
10. C. Goffman and G. Pedrick, *First Course in Functional Analysis*, Prentince-Hall, Englewood Cliffs 1965.
11. A. Grzegorczyk, On the definitions of computable real continuous functions, *Fund. Math.* **44** (1957) 61–71.
12. H. Heuser, *Funktionalanalysis*, B.G. Teubner, Stuttgart2. ed. 1986.
13. H. Ishihara, A constructive version of Banach's inverse mapping theorem, *New Zealand J. Math.* **23** (1994) 71–75.
14. H. Ishihara, Sequential continuity of linear mappings in constructive mathematics, *J. Univ. Comp. Sci.* **3** (1997) 1250–1254.
15. K.-I. Ko, *Complexity Theory of Real Functions*, Birkhäuser, Boston 1991.
16. S. Kutateladze, *Fundamentals of Functional Analysis*, Kluwer Academic Publishers, Dordrecht 1996.
17. D. Lacombe, Quelques procédés de définition en topologie récursive, in: A. Heyting (ed.), *Constructivity in mathematics*, North-Holland, Amsterdam 1959, 129–158.
18. G. Metakides, A. Nerode, and R. Shore, Recursive limits on the Hahn-Banach theorem, in: M. Rosenblatt (ed.), *Errett Bishop: Reflections on Him and His Research*, vol. 39 of *Contemporary Mathematics*, American Mathematical Society, Providence 1985, 85–91.
19. M. B. Pour-El and J. I. Richards, *Computability in Analysis and Physics*, Springer, Berlin 1989.
20. M. Schröder, Extended admissibility, *Theoret. Comp. Sci.* **284** (2002) 519–538.
21. M. Smyth, Topology, in: S. Abramsky, D. Gabbay, and T. Maibaum (eds.), *Handbook of Logic in Computer Science, Volume 1*, Clarendon Press, Oxford 1992, 641–761.
22. A. M. Turing, On computable numbers, with an application to the "Entscheidungsproblem", *Proc. London Math. Soc.* **42** (1936) 230–265.
23. M. Washihara, Computability and tempered distributions, *Mathematica Japonica* **50** (1999) 1–7.
24. K. Weihrauch, *Computable Analysis*, Springer, Berlin 2000.
25. K. Weihrauch and N. Zhong, Is the linear Schrödinger propagator Turing computable?, in: J. Blanck, V. Brattka, and P. Hertling (eds.), *Computability and Complexity in Analysis*, vol. 2064 of *Lect. Not. Comp. Sci.*, Springer, Berlin 2001, 369–377.
26. K. Weihrauch and N. Zhong, Is wave propagation computable or can wave computers beat the Turing machine?, *Proc. London Math. Soc.* **85** (2002) 312–332.
27. M. Yasugi, T. Mori, and Y. Tsujii, Effective properties of sets and functions in metric spaces with computability structure, *Theoret. Comp. Sci.* **219** (1999) 467–486.
28. N. Zhong, Computability structure of the Sobolev spaces and its applications, *Theoret. Comp. Sci.* **219** (1999) 487–510.

Appendix: The Remaining Proofs

The next lemma will be used for the following proof of Theorem 10.

Lemma 22. *Let (X, d, α) be a computable metric space. There exists a computable multi-valued operation $R :\subseteq X \times \mathcal{O}(X) \rightrightarrows \mathbb{N}$ such that for any open $U \subseteq X$ and $x \in U$ there exists some $k \in R(x, U)$ and $B(x, \overline{k}) \subseteq U$ holds for all such k.*

Proof. Given a sequence $\langle n_i, k_i\rangle_{i\in\mathbb{N}}$ of natural numbers such that

$$U = \bigcup_{i=0}^{\infty} B(\alpha(n_i), \overline{k_i})$$

and a sequence $(m_i)_{i\in\mathbb{N}}$ such that $d(\alpha(m_i), \alpha(m_j)) \leq 2^{-j}$ for all $i > j$ and $x := \lim_{i\to\infty} \alpha(m_i) \in U$, there exist $i, j \in \mathbb{N}$ such that $d(\alpha(n_i), \alpha(m_j))+2^{-j} < \overline{k_i}$ and thus we can effectively find $i, j, k \in \mathbb{N}$ such that $d(\alpha(n_i), \alpha(m_j))+2^{-j}+\overline{k} < \overline{k_i}$ and $\overline{k} > 0$. Then $d(x, y) < \overline{k}$ implies

$$\begin{aligned} d(\alpha(n_i), y) &\leq d(\alpha(n_i), \alpha(m_j)) + d(\alpha(m_j), x) + d(x, y) \\ &< d(\alpha(n_i), \alpha(m_j)) + 2^{-j} + \overline{k} \\ &< \overline{k_i} \end{aligned}$$

for all $y \in X$ and thus $B(x, \overline{k}) \subseteq B(\alpha(n_i), \overline{k_i}) \subseteq U$. □

Proof of Theorem 10

We consider the computable normed spaces $(X, ||\ ||, e)$ and $(Y, ||\ ||', e')$ with the dense sequences $\alpha := \alpha_e : \mathbb{N} \to X$, $\beta := \alpha_{e'} : \mathbb{N} \to Y$ according to Definition 5. Since no confusion is to be expected, we will also write $||\ ||$ instead of $||\ ||'$.

"(1)$\Longrightarrow$(2)" If T is open, then $\mathcal{O}(T)$ is well-defined and we have to prove that $\mathcal{O}(T)$ is computable if T is computable. We separate the proof into two parts (a) and (b). In (a) we use the fact that T is linear and open and in (b) we use the fact that T is computable.

(a) We prove that there exists a computable operation $R :\subseteq X \times \mathcal{O}(X) \rightrightarrows \mathbb{N}$ such that for any open $U \subseteq X$ and $x \in U$ there exists some $k \in R(x, U)$ and $B(Tx, \overline{k}) \subseteq T(U)$ and $\overline{k} > 0$ for all such k. Since T is open and linear, there exists some rational $r > 0$ such that $B(0, r) \subseteq T(B(0, 1))$. Given $U \in \mathcal{O}(X)$ and $x \in U$ we can effectively find some $n \in \mathbb{N}$ with $\varepsilon := \overline{n} > 0$ such that $B(x, \varepsilon) \subseteq U$ by Lemma 22 and some $k \in \mathbb{N}$ with $\overline{k} = \varepsilon r$. It follows by linearity of T

$$B(Tx, \overline{k}) = \varepsilon \left(B(0, r) + \frac{1}{\varepsilon} Tx \right) \subseteq \varepsilon \left(TB(0, 1) + \frac{1}{\varepsilon} Tx \right) = TB(x, \varepsilon) \subseteq T(U).$$

Thus, there exists a Turing machine M which computes a realization of R.

(b) Let M' be a Turing machine which computes a (δ_X, δ_Y)–realization of T. We will construct a Turing machine M'' which computes a $(\delta_{\mathcal{O}(X)}, \delta_{\mathcal{O}(Y)})$–realization of $\mathcal{O}(T)$. The set

$$W := \{01^{n_0+1}...01^{n_l+1}0 \in \Sigma^* : l \in \mathbb{N} \text{ and } ||\alpha(n_i) - \alpha(n_j)|| < 2^i \text{ for } i < j \leq l\}$$

is an r.e. subset of Σ^* with $\delta_X(W\Sigma^\omega) = X$. Let $p \in \text{dom}(\delta_{\mathcal{O}(X)})$ with $U := \delta_{\mathcal{O}(X)}(p)$.

Now machine M'' on input p searches systematically for some finite word $w = 01^{n_0+1}01^{n_1+1}...01^{n_i+1}0 \in W$ such that machine M with input $\langle w0^\omega, p\rangle$ produces some (encoded) output $m \in \mathbb{N}$ and M' with input $w0^\omega$ some output v, both after reading only w or some finite prefix of it, such that the following holds: if $01^{k_0+1}01^{k_1+1}...01^{k_j+1}0$ is the longest prefix of v which ends with 0, then $2^{-j+2} < \overline{m}$ and $\alpha(n_i) \in U$. Whenever machine M'' finds such a word w, then it writes $01^{\langle k_j,k\rangle+1}$ with $\overline{k} = 2^{-j+1}$ on the output tape.

If this happens, then $B(Tx, \overline{m}) \subseteq T(U)$ and $||y - Tx|| \leq 2^{-j}$ with $x := \alpha(n_i)$ and $y := \beta(k_j)$. Thus, $||z - y|| < \overline{k}$ implies

$$||z - Tx|| \leq ||z - y|| + ||y - Tx|| < \overline{k} + 2^{-j} < \overline{m}$$

for all $z \in Y$ and thus $B(\beta(k_j), \overline{k}) \subseteq B(Tx, \overline{m}) \subseteq T(U)$. Hence, we obtain $\delta_{\mathcal{O}(Y)}(q) \subseteq T(U)$ for the output q of M'', provided that this output q is infinite.

It remains to prove that M'' on input p actually produces an infinite output q and that $T(U) \subseteq \delta_{\mathcal{O}(Y)}(q)$. Thus, let $y \in T(U)$. Then there is some $x \in U$ with $Tx = y$ and some $r \in \text{dom}(\delta_X)$ with $\delta_X(r) = x$ such that $r = 01^{n_0+1}01^{n_1+1}...$ has infinitely many prefices in W and $\alpha(n_i) \in U$ for all i; especially there is one such prefix $w' \in W$ of r such that M on input $\langle w'0^\omega, p\rangle$ stops with output $m \in \mathbb{N}$ while reading w' or some finite prefix of it and there is some prefix $w \in W$ of r which is longer than w' such that M' on input $w0^\omega$ writes some output $01^{k_0+1}01^{k_1+1}...01^{k_j+1}0$ with $2^{-j+2} < \overline{m}$ while reading w or some finite prefix of it. Finally, M'' will find such a word w and write $01^{\langle k_j,k\rangle+1}$ with $\overline{k} = 2^{-j+1}$ on the output tape. We obtain $y \in B(\beta(k_j), \overline{k})$ since $||\beta(k_j) - y|| \leq 2^{-j} < \overline{k}$. Moreover, M'' will find infinitely many such words w and produce an infinite output q with $T(U) \subseteq \delta_{\mathcal{O}(Y)}$.

"(2)$\Longrightarrow$(1)" Now let $\mathcal{O}(T)$ be well-defined and computable and let M be a Turing machine which computes a realization of $\mathcal{O}(T)$. Then T is open since $\mathcal{O}(T)$ is well-defined. Since T is bounded, there exists a rational bound $s > 0$ such that $||Tx|| \leq s||x||$ for all $x \in X$ and some $j \in \mathbb{N}$ such that $2^j > s$. We construct a Turing machine M' which computes a (δ_X, δ_Y)–realization of T. Given some input $p = 01^{n_0+1}01^{n_1+1}01^{n_2+1}0...$ with $x := \delta_X(p)$, machine M' works in steps $i = 0, 1, 2, ...$ as follows: in step i machine M' starts machine M with input $q = 01^{\langle n_{i+j+2},k_i\rangle+1}01^{\langle n_{i+j+2},k_i\rangle+1}01^{\langle n_{i+j+2},k_i\rangle+1}0...$ where $\overline{k_i} := 2^{-i-j-2}$ and simulates M until it writes the first word $01^{\langle n,k\rangle+1}0$. Then M' writes 01^{n+1} on its output tape and continues with the next step $i + 1$.

Since $\delta_{\mathcal{O}(X)}(q) = B(\alpha(n_{i+j+2}), \overline{k_i})$, M produces an output r with

$$\delta_{\mathcal{O}(Y)}(r) = \mathcal{O}(T)(\delta_{\mathcal{O}(X)}(q)) = TB(\alpha(n_{i+j+2}), \overline{k_i}).$$

Thus, for any subword 01^{n+1} which is written by M' in step i on its output tape, we obtain $\beta(n) \in TB(\alpha(n_{i+j+2}), \overline{k_i})$. Since $||x - \alpha(n_{i+j+2})|| \leq \overline{k_i}$, it follows

$$||\beta(n) - Tx|| \leq ||\beta(n) - T\alpha(n_{i+j+2})|| + ||T\alpha(n_{i+j+2}) - Tx|| \leq 2s\overline{k_i} < 2^{-i-1}$$

and hence $\delta_Y(t) = Tx$ holds for the infinite output t of M'. □

The next lemma will be used for the following proof of Proposition 13.

Lemma 23. *There exists a computable sequence $(b_n)_{n\in\mathbb{N}}$ of positive right-computable real numbers such that for any computable function $f : \mathbb{N} \to \mathbb{N}$ there exists some $i \in \mathbb{N}$ such that $2^{-f(i)} > b_i$.*

Proof. We use some total Gödel numbering $\varphi : \mathbb{N} \to P$ of the set of partial computable functions $P := \{f :\subseteq \mathbb{N} \to \mathbb{N} : f \text{ computable}\}$ to define

$$b_n := \begin{cases} 2^{-k-1} & \text{if } \varphi_n(n) = k \\ 1 & \text{if } \varphi_n(n) \text{ is undefined} \end{cases}$$

for all $n, k \in \mathbb{N}$. Then $(b_n)_{n\in\mathbb{N}}$ is a computable sequence of positive right-computable real numbers. Now let $f : \mathbb{N} \to \mathbb{N}$ be some total computable function. Then there exists some $i \in \mathbb{N}$ such that $\varphi_i = f$ and we obtain $2^{-f(i)} > 2^{-\varphi_i(i)-1} = b_i$. □

Proof of Proposition 13

We sketch the proof. For a complete version see [4]. Given the computable sequence $(b_n)_{n\in\mathbb{N}}$ from Lemma 23 we can effectively determine a sequence $(T_n)_{n\in\mathbb{N}}$ of diagonal operator $T_n : \ell_2 \to \ell_2$ as follows: for any b_n we effectively determine a decreasing sequence $(a_{nk})_{k\in\mathbb{N}}$ of rational numbers $a_{nk} \in \mathbb{Q}$ such that $a_{n0} = 1$ and $b_n = \inf_{k\in\mathbb{N}} a_{nk}$. Now we define $T_n : \ell_2 \to \ell_2$ by $T_n(x_k)_{k\in\mathbb{N}} := (a_{nk}x_k)_{k\in\mathbb{N}}$ for all $(x_k)_{k\in\mathbb{N}} \in \ell_2$. Given some $x = (x_k)_{k\in\mathbb{N}}$ and a precision $m \in \mathbb{N}$ we can effectively find some $j \in \mathbb{N}$ and numbers $q_0, ..., q_j \in Q_{\mathbb{F}}$ such that $||\sum_{i=0}^{j} q_i e_i - x||_2 < 2^{-m}$. Since $||T_n|| = 1$ as one can easily see, it follows

$$\left|\left|T_n\left(\sum_{i=0}^{j} q_i e_i\right) - T_n(x)\right|\right|_2 \leq ||T_n|| \cdot \left|\left|\sum_{i=0}^{j} q_i e_i - x\right|\right|_2 < 2^{-m}$$

By linearity of T_n we obtain $T_n(\sum_{i=0}^{j} q_i e_i) = \sum_{i=0}^{j} q_i T_n(e_i) = \sum_{i=0}^{j} q_i a_{ni}$ and thus we can evaluate T_n effectively up to any given precision m. Using type conversion we can prove that given the sequence $(b_n)_{n\in\mathbb{N}}$, we can actually find the sequence $(T_n)_{n\in\mathbb{N}}$ effectively. Thus, since $(b_n)_{n\in\mathbb{N}}$ is computable, it follows that $(T_n)_{n\in\mathbb{N}}$ is a computable sequence in $\mathcal{C}(\ell_2, \ell_2)$. Now, if the sequence $(T_nB(0,1))_{n\in\mathbb{N}}$ would be computable, then the computable operation $R :\subseteq \ell_2 \times \mathcal{O}(\ell_2) \rightrightarrows \mathbb{N}$ from Lemma 22 would yield a computable sequence $(r_n)_{n\in\mathbb{N}}$ of rationals with $r_n \in R(0, T_nB(0,1))$ such that $0 < r_n \leq b_n$. But this contradicts Lemma 23. □

Proof of Theorem 15

"(2)$\Longrightarrow$(1)" We consider the computable metric spaces (X, d, α) and (Y, d', β). We note that T is continuous, if $\mathcal{O}(T^{-1})$ is well-defined. Given a Turing machine M which computes a realization of $\mathcal{O}(T^{-1})$, we construct a Turing machine M' which computes a realization of T. The machine M' with input $p \in \mathrm{dom}(\delta_X)$ works in steps $k = 0, 1, 2, ...$ as follows. In step k machine M' simultaneously tests all values $n \in \mathbb{N}$ until some value is found with the following property: machine M with input $01^{\langle n,m\rangle+1}01^{\langle n,m\rangle+1}01^{\langle n,m\rangle+1}0...$ with $\overline{m} = 2^{-k}$ produces an output with subword $01^{\langle i,j\rangle+1}0$ such that $x = \delta_X(p) \in B(\alpha(i), \overline{j})$. As soon as such a subword is found, M' writes 01^{n+1} on the output tape.

If this happens, i.e. if M' writes 01^{n+1} on its output tape, then we obtain $Tx \in B(\beta(n), 2^{-k})$ since $x \in B(\alpha(i), \overline{j}) \subseteq T^{-1}(B(\beta(n), \overline{m}))$. Moreover, M' actually produces an infinite output q, since for any $k \in \mathbb{N}$ there is some $n \in \mathbb{N}$ such that $Tx \in B(\beta(n), 2^{-k})$ and thus $x \in T^{-1}(B(\beta(n), 2^{-k}))$ and consequently M on input $01^{\langle n,m\rangle+1}01^{\langle n,m\rangle+1}0...$ has to produce some output with subword $01^{\langle i,j\rangle+1}0$ and $x \in B(\alpha(i), \overline{j})$. It follows $\delta_Y(q) = Tx$. □