

A GALOIS CONNECTION BETWEEN TURING JUMPS AND LIMITS

VASCO BRATTKA

Faculty of Computer Science, Universität der Bundeswehr München, Germany and Department of Mathematics and Applied Mathematics, University of Cape Town, South Africa
e-mail address: Vasco.Brattka@cca-net.de

ABSTRACT. Limit computable functions can be characterized by Turing jumps on the input side or limits on the output side. As a monad of this pair of adjoint operations we obtain a problem that characterizes the low functions and dually to this another problem that characterizes the functions that are computable relative to the halting problem. Correspondingly, these two classes are the largest classes of functions that can be pre or post composed to limit computable functions without leaving the class of limit computable functions. We transfer these observations to the lattice of represented spaces where it leads to a formal Galois connection. We also formulate a version of this result for computable metric spaces. Limit computability and computability relative to the halting problem are notions that coincide for points and sequences, but even restricted to continuous functions the former class is strictly larger than the latter. On computable metric spaces we can characterize the functions that are computable relative to the halting problem as those functions that are limit computable with a modulus of continuity that is computable relative to the halting problem. As a consequence of this result we obtain, for instance, that Lipschitz continuous functions that are limit computable are automatically computable relative to the halting problem. We also discuss 1-generic points as the canonical points of continuity of limit computable functions, and we prove that restricted to these points limit computable functions are computable relative to the halting problem. Finally, we demonstrate how these results can be applied in computable analysis.

1. INTRODUCTION

Limit computable functions have been studied for a long time. In computability theory limits appear, for instance, in the form of Shoenfield’s limit lemma [46]. In algorithmic learning theory they have been introduced by Gold [22]. Later, limit computable functions have been studied by Wagner [50, 51], who calls them *Turing operators of the first kind* (and he attributes this class to Freivald [18]). Wagner discusses composition and also normal form theorems. In computable analysis limit computations were studied by Freund [19] who introduced Turing machines that can revise their output as well as non-deterministic Turing machines. Later, similar machines were systematically studied by Ziegler [56, 55]. Limit computable real numbers were analyzed by Freund and Staiger [20] and by Zheng

Key words and phrases: Computable analysis, limit computability, computability relative to the halting problem, lowness, 1-genericity, represented spaces, Galois connections.

and Weihrauch [53] and others. On the other hand, Ho [25, 26] studied functions in analysis that are computable relative to the halting problem.

Many of these results are scattered in the literature, and it remains somewhat unclear how all these results are related. Some clarity can be brought into the picture if one starts with the notion of a limit computable function and develops it systematically. It turns out that at the heart of such a development there is an adjoint situation that is related to the fact that limit computations can either be described with limits on the output side or with Turing jumps on the input side. Shoenfield’s limit lemma is the non-uniform correlate of this observation. From this perspective functions computable relative to the halting problem and low functions are natural notions that should be studied alongside with limit computable functions, since they constitute the largest classes of functions that can be post or pre composed with limit computable functions without leaving the class of limit computable functions. We will develop these notions systematically in sections 2 and 3.¹ Together with these notions we also study 1-generic points, as these are the canonical points of continuity of limit computable functions.

In section 4 we transfer these results to the setting of represented spaces and hence to other data types. The crucial notion here is the notion of a jump of a representation that was introduced by Ziegler [56, 55] and further generalized by de Brecht [15]. On the level of represented spaces the adjoint situation between limits and Turing jumps can formally be expressed as a Galois connection. We also study how the jump of a represented space interacts with other constructions on represented spaces such as building products and exponentials.

In section 5 we transfer our results to the setting of computable metric spaces. The limit normal form for limit computable functions can easily be expressed using limits in computable metric spaces and for the jump normal form one can use a general jump operation on computable metric spaces. We also transfer a characterization of 1-generic points to the level of computable metric spaces.

In section 6 we continue to study mostly computable metric spaces and specifically the relation between limit computable functions and functions computable relative to the halting problem. One characterization shows that functions that are computable relative to the halting problem are exactly those continuous limit computable functions that admit a global modulus of continuity that is computable relative to the halting problem. We also provide several examples of functions of different type that show that not all continuous limit computable functions are computable relative to the halting problem. However, we can provide oracle classes that are sufficient to compute limit computable functions that are continuous, uniformly continuous and Lipschitz continuous, respectively. In particular for Lipschitz continuous functions limit computability and computability with respect to the halting problem coincide.

We close this article with section 7 in which we demonstrate how some known results can easily be derived using the techniques provided in this article. We also include some simple new applications. The discussed examples also highlight connections between results that are seemingly unrelated or scattered in other sources.

¹We note that an unpublished draft of some of the material presented here was circulated since 2007 and it has influenced some of the references, such as joint work by the author with de Brecht and Pauly [7]. The material presented in section 3 and everything based on it has only been developed recently.

2. LIMIT COMPUTABILITY

In this section we are going to characterize limit computable functions in different ways. We start with recalling the definition. Limit computable functions are defined on Turing machines that have a two-way output tape, and these machines are allowed to revise their output.

Definition 2.1 (Limit computable functions). A function $F : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ is called *limit computable*, if there exists a Turing machine M that upon any input $p \in \text{dom}(F)$ computes $F(p)$ on its two-way output tape in the long run such that on any output position $n \in \mathbb{N}$ after a finite number of steps the output is $F(p)(n)$ and will not be changed anymore.

If the Turing machine does only change the entire output (over all $n \in \mathbb{N}$) finitely many times for each fixed input p , then F is called *computable with finitely many mind changes*. This is a strengthening of the notion of limit computability.

It is clear that all computable maps are limit computable and limit computable maps are obviously closed under restriction. The additional flexibility to revise the output tape gives additional power to limit machines that ordinary Turing machines do not have. As examples we mention some discontinuous functions that are not computable, but limit computable. For the definition we use tupling functions. We define $\langle n, k \rangle := \frac{1}{2}(n+k)(n+k+1) + k$, $\langle p, q \rangle(2n) := p(n)$ and $\langle p, q \rangle(2n+1) := q(n)$, $\langle p_0, p_1, p_2, \dots \rangle \langle n, k \rangle := p_n(k)$, and $\langle n, p \rangle := np$ for all $p, q, p_i \in \mathbb{N}^{\mathbb{N}}$ and $n, k \in \mathbb{N}$. By $\hat{n} = nnn\dots \in \mathbb{N}^{\mathbb{N}}$ we denote the constant sequence with value $n \in \mathbb{N}$.

Example 2.2 (Limit computable maps). The following maps are limit computable but not continuous and hence not computable:

(1) The *equality test for zero*

$$E : \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}, p \mapsto \begin{cases} \hat{1} & \text{if } p = \hat{0} \\ \hat{0} & \text{otherwise} \end{cases}$$

(2) The *limit map*

$$\lim : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}, \langle p_0, p_1, p_2, \dots \rangle \mapsto \lim_{i \rightarrow \infty} p_i.$$

(3) The *Turing jump* $J : \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ by

$$J(p)(i) := \begin{cases} 1 & \text{if Turing machine } i \text{ halts upon input } p \\ 0 & \text{otherwise} \end{cases}$$

for all $p \in \mathbb{N}^{\mathbb{N}}$ and $i \in \mathbb{N}$.

The function E is even computable with finitely many mind changes.

By \lim_{Δ} we denote the restriction of \lim to convergent sequences with respect to the discrete topology on $\mathbb{N}^{\mathbb{N}}$. In some sense one can say that the limit map and J are not only some limit computable maps, but prototypes of limit computable maps. Before we can prove a formal version of this statement, we first discuss compositions of computable and limit computable maps that turn out to be limit computable.

Proposition 2.3 (Composition). *If $F : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ is limit computable and $G : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ is computable, then $F \circ G$ and $G \circ F$ are both limit computable.*

Proof. Let M_F be a limit machine computing F and M_G an ordinary Turing machine computing G . Then a limit machine M computing $F \circ G$ can just be obtained by composing the two machines M_F and M_G in the straightforward way.

A limit machine M' for $G \circ F$ can also be constructed by composing the machines M_F and M_G . However, the composition has to be done such that if M_F changes the content of some output cell n , then the computation of M_G has to be restarted at time step n of its program (which is sufficient to guarantee that M_G has not yet seen the content of cell n). Since the output of M_F of any finite length eventually stabilizes and M_G uses a one-way output tape, it follows that M' produces a converging output in this way for any input $p \in \text{dom}(G \circ F)$. \square

It is easy to see that the first statement of the proposition can even be strengthened in the following way.

Proposition 2.4 (Composition with finitely many mind changes). *If $F : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ is limit computable and $G : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ is computable with finitely many mind changes, then $F \circ G$ is limit computable.*

For the composition $G \circ F$ of functions as in Proposition 2.4 the strategy of the corresponding proof of Proposition 2.3 does not work in general. In fact, it is not too difficult to construct a counterexample.

Example 2.5 (Composition). The equality test E is computable with finitely many mind changes, and the limit map lim is limit computable, but $E \circ \text{lim}$ is not limit computable.

We leave the proof to the reader. Now we provide a very useful characterization of limit computable functions. This result is also implicit in the work of Wagner [50].

Theorem 2.6 (Limit normal form). *A function $G : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ is limit computable if and only if there exists a computable function $F : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$, such that $G = \text{lim} \circ F$.*

Proof. By Example 2.2 lim is limit computable. It follows from Proposition 2.3 that if F is computable, then $\text{lim} \circ F$ is limit computable.

Conversely, if G is limit computable, then we construct a one-way output Turing machine M_F that computes a function $F : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$. This machine M_F internally simulates a limit machine M_G for G on input p . The machine M_F successively produces an output $\langle q_0, q_1, \dots \rangle$ in steps $\langle i, j \rangle = 0, 1, 2, \dots$. In step $\langle i, j \rangle$ machine M_F simulates M_G for i time steps beyond the point where M_G has filled its j -th output position for the first time. Machine M_F writes the resulting content of the simulated output of M_G in the j -th position into its own output $q_i(j)$. If the simulated j -th position eventually stabilizes, then $\lim_{i \rightarrow \infty} q_i(j)$ exists and coincides with the final value of the j -th position $G(p)(j)$. Thus the machine M_F computes a function $F : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ with $G = \text{lim} \circ F$. Here F is restricted to $\text{dom}(G)$. \square

We note that the limit normal form theorem allows us to study limit computable functions in terms of ordinary computable functions and without considering limit machines. The particular output behavior of limit machines can be expressed directly using the limit map. We also obtain the following characterization of limit computable functions as pointwise limits of a sequence of computable functions.

Corollary 2.7 (Pointwise limit). *A function $G : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ is limit computable if and only if there is a computable sequence $(F_n)_{n \in \mathbb{N}}$ of computable functions $F_n : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ with $\text{dom}(G) \subseteq \text{dom}(F_n)$ and such that $\lim_{n \rightarrow \infty} F_n(p) = G(p)$ for all $p \in \text{dom}(G)$.*

Theorem 2.6 together with Proposition 2.3 yield the following corollary.

Corollary 2.8. *For any computable function $F : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ there exists a computable function $G : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ such that $\lim \circ G = F \circ \lim$.*

Corollary 2.8 also holds with continuity instead of computability in both occurrences. We note that the computable functions F do not constitute the largest class of functions for which there is a computable G with $\lim \circ G = F \circ \lim$. In Corollary 3.5 we extend this result to functions that are computable relative to the halting problem. Maps that satisfy the same property as the limit map in Corollary 2.8 have been called *jump operators* [7, 15] or *transparent* [9].

Definition 2.9 (Transparency). A function $T : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ is called *transparent* if for every computable $F : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ there exists a computable function $G : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ such that $T \circ G = F \circ T$.

It is easy to see that not all functions are transparent, but the class of transparent functions is reasonably large.

Proposition 2.10 (Transparency). *The class of transparent functions $T : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ forms a monoid with respect to composition, i.e., the identity is transparent, and transparent functions are closed under composition.*

It is clear that limit computable maps do not necessarily map computable inputs to computable outputs. However, they map computable inputs to limit computable outputs.

Definition 2.11 (Limit computable points). A point $p \in \mathbb{N}^{\mathbb{N}}$ is called *limit computable*, if there exists a computable sequence $(p_i)_{i \in \mathbb{N}}$ in $\mathbb{N}^{\mathbb{N}}$ such that $p = \lim_{i \rightarrow \infty} p_i$.

It follows directly from the limit normal form theorem (Theorem 2.6) and the fact that computable maps map computable inputs to computable outputs that limit computable maps map computable inputs to limit computable outputs.

Corollary 2.12. *If $F : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ is limit computable and $p \in \text{dom}(F)$ is computable, then $F(p)$ is limit computable.*

It also follows from the limit normal form theorem (Theorem 2.6) that a point $p \in \mathbb{N}^{\mathbb{N}}$ is limit computable if and only if it is the value of a constant limit computable map $c : \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$.

We now study the Turing jump J , and we first prove that its inverse is computable. Even though J is not continuous, it is injective and has a computable inverse.

Proposition 2.13 (Jump inversion). *The Turing jump operator J is injective, and its partial inverse $J^{-1} : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ is computable.*

Proof. There exists a computable function $r : \mathbb{N} \rightarrow \mathbb{N}$ such that the Turing machine with code $r\langle n, k \rangle$ halts upon input $p \in \mathbb{N}^{\mathbb{N}}$ if and only if $p(n) = k$. The function $F : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ defined by

$$F(q)(n) := \min\{k \in \mathbb{N} : qr\langle n, k \rangle = 1\}$$

for all $q \in \mathbb{N}^{\mathbb{N}}$, $n \in \mathbb{N}$ and $\text{dom}(F) := \{q \in \mathbb{N}^{\mathbb{N}} : (\forall n)(\exists k) qr\langle n, k \rangle = 1\}$ is computable. We obtain

$$F \circ J(p)(n) = \min\{k \in \mathbb{N} : J(p)(r\langle n, k \rangle) = 1\} = \min\{k \in \mathbb{N} : p(n) = k\} = p(n)$$

for all $p \in \mathbb{N}^{\mathbb{N}}$ and $n \in \mathbb{N}$, i.e., $F \circ J = \text{id}$. This implies that J is injective and $J^{-1} = F|_{\text{range}(J)}$ is computable. \square

We also write $p' := J(p)$, and we call p' the *Turing jump* of p , and we call $0' := J(\widehat{0})$ the *halting problem*. We note that the Turing jump operator considered as a map on Turing degrees is not injective.² We mention that Proposition 2.13 also implies that any point can be reduced to its Turing jump, i.e., $p \leq_T p'$ (of course, this reduction is known to be strict, which can be proved by an easy diagonalization argument).

It is perhaps surprising that there is a dual version of the limit normal form theorem that characterizes limit computation by an input modification with the help of Turing jumps instead of an output modification with limits. This shows that in some sense Turing jumps and topological limits are adjoint to each other. An analogous result can be found in the work of Wagner [50].

Theorem 2.14 (Jump normal form). *A function $F : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ is limit computable, if and only if there exists a computable function $G : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ such that $F = G \circ J$.*

Proof. By Example 2.2 the Turing jump operator $J : \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ is limit computable. Hence for any computable $G : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ the composition $G \circ J$ is limit computable by Proposition 2.3.

Let now M_F be a limit Turing machine that computes $F : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$. We describe a Turing machine M_G that computes a function $G : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ with $F(p) = G \circ J(p)$ for all $p \in \text{dom}(F)$. There exists a computable function $r : \mathbb{N} \rightarrow \mathbb{N}$ such that the Turing machine with code $r\langle n, t \rangle$ halts upon input p if and only if M_F on input p changes the output cell n after more than t steps. Now M_G on input q works as follows: it simulates the machine M_F on input $p = J^{-1}(q)$, where J^{-1} is the partial inverse of the Turing jump operator that is computable by Proposition 2.13, and writes the output to some working tape. After simulating M_F for t steps at most the first k cells on this working tape have been used for some $k \in \mathbb{N}$. In this situation M_G checks $qr\langle n, t \rangle$ for all $n = 0, \dots, k$ and when the result is 0 for some initial segment $n = 0, \dots, j$, then M_G copies the content of those cells $n = 0, \dots, j$ to the output tape that have not yet been written to the output. This algorithm ensures that only those output cells which have already stabilized are copied to the output. Thus, M_G operates with a one-way output tape and computes a function $G : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ with $F(p) = G \circ J(p)$ for all $p \in \text{dom}(F)$. Since J is injective, one can restrict G such that one obtains $F = G \circ J$. \square

We obtain the following corollary, which is somehow dual to the statement of Corollary 2.8.

Corollary 2.15. *For any computable function $F : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ there exists a computable function $G : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ such that $J \circ F = G \circ J$.*

It is clear that J and lim cannot be swapped in Corollaries 2.8 and 2.15. In particular, J is not transparent, since it has no computable values in its range. However, it is not too difficult to see that J^{-1} is transparent.

It is a consequence of Corollary 2.15 that the Turing jump operator is monotone with respect to Turing reducibility.

Corollary 2.16 (Monotonicity of Turing jumps). *$q \leq_T p \implies q' \leq_T p'$ for all $p, q \in \mathbb{N}^{\mathbb{N}}$.*

We also mention that the jump normal form theorem (Theorem 2.14) yields as a non-uniform corollary a version of Shoenfield's limit lemma [46].

²There are $p \not\equiv_T q'$ such that $p' \equiv_T q'$.

Corollary 2.17 (Shoenfield’s limit lemma 1959). *A point $p \in \mathbb{N}^{\mathbb{N}}$ is limit computable if and only if $p \leq_T 0'$.*

One can prove relativized forms of the jump normal form theorem (Theorem 2.14) and the limit normal form theorem (Theorem 2.6). However, the exact relativization needs some care. For every $q \in \mathbb{N}^{\mathbb{N}}$ we define the *relativized jump operator* $J_q : \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ by $J_q(p) := J\langle q, p \rangle$. We say that $F : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ is *limit computable relative to $q \in \mathbb{N}^{\mathbb{N}}$* if there exists a limit computable $G : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ such that $F(p) = G\langle q, p \rangle$ for all $p \in \text{dom}(F)$. Now the following theorem can be proved along the lines of the above results.

Theorem 2.18 (Relativized limit computability). *The following are equivalent for $q \in \mathbb{N}^{\mathbb{N}}$:*

- (1) F is limit computable relative to q ,
- (2) $F = \lim \circ G$ for some G that is computable relative to q ,
- (3) $F = G \circ J_q$ for some G that is computable.

The symmetry between the two normal forms does not fully extend to the relativized case since oracles act on the input side, and hence we need to use either functions that are computable relative to q or the relativized jump J_q . Related to this observation the equivalence expressed in the two normal form theorems is not fully uniform and, in fact, J^{-1} is not transparent in a topological sense (see [7, Proposition 9.13]).

It is rarely mentioned that 1-generic points as they are used in computability theory [47] can be characterized as points of continuity of the Turing jump J . This was noticed in [7, Lemma 9.3].

Definition 2.19 (1-generic). A point $p \in \mathbb{N}^{\mathbb{N}}$ is called *1-generic* if and only if J is continuous at p .

It is easy to see that 1-generic points cannot be computable. The following characterization of 1-generic points is a consequence of Theorem 2.14. The 1-generic points are exactly those at which every limit computable function is continuous.

Corollary 2.20 (1-generic points). *A point $p \in \mathbb{N}^{\mathbb{N}}$ is 1-generic if and only if every limit computable function $F : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ with $p \in \text{dom}(F)$ is continuous at p .*

While the “only if” direction is a consequence of Theorem 2.14, one obtains the “if” direction by applying the statement to $F = J$. We note that there are limit computable functions (such as the limit map \lim) that do not have any 1-generics in their domain, simply because they are not continuous at any point.

It is easy to see that restricted to 1-generics the Turing jump operator is continuous and computable relative to the halting problem.

Proposition 2.21 (Jump on 1-generics). *The Turing jump operator $J : \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}, p \mapsto p'$ restricted to the set of 1-generics is computable relative to the halting problem.*

Proof. The sets

- (1) $A := \{(w, i) \in \mathbb{N}^* \times \mathbb{N} : \text{the } i\text{-th Turing machine halts on all extensions of } w\}$,
- (2) $B := \{(w, i) \in \mathbb{N}^* \times \mathbb{N} : \text{the } i\text{-th Turing machine halts on no extension of } w\}$

are both computable relative to $0'$. Given a 1-generic $p \in \mathbb{N}^{\mathbb{N}}$ it is clear that for each $i \in \mathbb{N}$ there is a $w \sqsubseteq p$ such that $(w, i) \in A$ or $(w, i) \in B$. Depending on the answer we know that $J(p)(i) = 1$ or $J(p)(i) = 0$. Hence, J restricted to 1-generics is computable relative to the halting problem. \square

As a consequence of the jump normal form (Theorem 2.14) we obtain that every limit computable function is computable relative to the halting problem, when restricted to the 1-generic inputs.

Corollary 2.22 (Limit computability on 1-generics). *Restricted to 1-generics every limit computable $F : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ is computable relative to the halting problem.*

As a non-uniform corollary of Proposition 2.21 we obtain that all 1-generics are *generalized low*, a property that is made more precise in the next well-known corollary [28, Lemma 2].

Corollary 2.23 (Jockusch 1977). $p' \equiv_{\text{T}} \langle p, 0' \rangle$ for all 1-generics $p \in \mathbb{N}^{\mathbb{N}}$.

Here the reduction $p' \leq_{\text{T}} \langle p, 0' \rangle$ follows directly from Proposition 2.21, and the inverse reduction even holds for arbitrary $p \in \mathbb{N}^{\mathbb{N}}$.

The points whose jump is below $0'$ also have a special name, they are called *low*.

Definition 2.24 (Low points). A point $p \in \mathbb{N}^{\mathbb{N}}$ is called *low*, if its Turing jump p' is limit computable.

It is clear that all computable points are low, and all limit computable 1-generics are low by Corollary 2.23. One can use this observation to show that the class of low points also contains non-computable points. Using the *low map* $L := J^{-1} \circ \text{lim}$ we obtain the following characterization of low points [7, Lemma 8.2].

Corollary 2.25 (Low points). *A point $p \in \mathbb{N}^{\mathbb{N}}$ is low if and only if there is a computable $q \in \mathbb{N}^{\mathbb{N}}$ such that $L(q) = p$.*

It follows from Corollaries 2.16 and 2.17 that computable functions map low inputs to low outputs. In [7] low functions were introduced that were further studied in [9, 11].

Definition 2.26 (Low functions). A function $F : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ is called *low* if there is a computable $G : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ such that $F = L \circ G$.

This definition captures the idea that the result of F is computed as a low point. It is clear that low functions map computable inputs to low outputs. We obtain the following closure properties under composition.

Proposition 2.27 (Composition with low functions). *Let $F, G : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ be functions. If F and G are low then so is $G \circ F$. If G is limit computable and F is low, then $G \circ F$ is limit computable.*

Proof. Both observations are based on the jump normal form theorem (Theorem 2.14) that yields a computable $R : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ with $\text{lim} = R \circ J$. By Corollary 2.8 we also have a computable $S : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ with $R \circ \text{lim} = \text{lim} \circ S$. This implies $L \circ L = J^{-1} \circ \text{lim} \circ J^{-1} \circ \text{lim} = J^{-1} \circ R \circ J \circ J^{-1} \circ \text{lim} = L \circ S$ and $\text{lim} \circ L = R \circ J \circ J^{-1} \circ \text{lim} = R \circ \text{lim}$. The cases of the composition of general low and limit computable functions can easily be derived from these observations. \square

In fact, the low functions form the largest class of functions that can be composed with limit computable functions from the left without leaving the class of limit computable functions (see also [11, Proposition 14.16] for a related result).

Corollary 2.28 (Low functions). *A function $F : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ is low if and only if $G \circ F$ is limit computable for every limit computable $G : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$.*

The “only if” direction is a direct consequence of Proposition 2.27. The “if” direction follows from the limit normal form theorem if one applies the assumption to $G = J$. As a consequence of Proposition 2.4 we get the following corollary.

Corollary 2.29 (Finite mind changes are low). *Every function $F : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ that is computable with finitely many mind changes is low.*

Hence, in contrast to low points $p \in \mathbb{N}^{\mathbb{N}}$, we have a lot of natural examples of low functions, such as the equality test E .

We close this section by mentioning briefly that there is a characterization of functions that are computable with finitely many mind changes that is analogous to Theorem 2.6.

Theorem 2.30 (Discrete limit normal form). *A function $G : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ is computable with finitely many mind changes if and only if there exists a computable function $F : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$, such that $G = \lim_{\Delta} \circ F$.*

We leave the simple proof to the reader (see [7]).

3. COMPUTABILITY RELATIVE TO THE HALTING PROBLEM

Now we consider functions that are computable relative to the halting problem $0'$. It turns out that they play a dual rôle to the low functions that are characterized in terms of $L = J^{-1} \circ \lim$. The functions that are computable relative to the halting problem can be characterized in terms of $H := J \circ \lim^{-1} : \mathbb{N}^{\mathbb{N}} \rightrightarrows \mathbb{N}^{\mathbb{N}}$.

The first result shows that H is computable relative to the halting problem. In order to prove this we need to invert limits such that we obtain control on the jump of the resulting converging sequence. Roughly speaking, this is possible since there are many sequences converging to a given point, and this gives us enough freedom to choose a suitable one. We recall that the *composition* of two multi-valued functions $f : \subseteq X \rightrightarrows Y$ and $g : \subseteq Y \rightrightarrows Z$ is defined by $(g \circ f)(x) := \{z \in Z : (\exists y \in f(x)) z \in g(y)\}$ with $\text{dom}(g \circ f) := \{x \in X : x \in \text{dom}(f) \text{ and } f(x) \subseteq \text{dom}(g)\}$. A multi-valued $f : \subseteq \mathbb{N}^{\mathbb{N}} \rightrightarrows \mathbb{N}^{\mathbb{N}}$ is called *computable (relative to some oracle q)* if there is a single-valued $F : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ with the corresponding property and such that $F(p) \subseteq f(p)$ for all $p \in \text{dom}(f)$ and $\text{dom}(f) \subseteq \text{dom}(F)$. We recall that a function $F : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ is called *computable relative to some oracle $q \in \mathbb{N}^{\mathbb{N}}$* if there is some computable function $G : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ such that $F(p) = G\langle q, p \rangle$ for all $p \in \text{dom}(F)$. In terms of computability theory, the next proof uses the *finite extension method*.

Theorem 3.1 (Limit inversion). *$H = J \circ \lim^{-1}$ is computable relative to the halting problem.*

Proof. We need to show that there exists a function $I : \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ with $\lim \circ I = \text{id}$ and such that $J \circ I$ is computable relative to $0'$. Given $p \in \mathbb{N}^{\mathbb{N}}$ we describe the computation of I relative to $0'$ by an inductive construction in $i \in \mathbb{N}$. We start with $w_0 := \varepsilon \in \mathbb{N}^*$ and $t_0 := 0 \in \mathbb{N}$. Given a finite sequence of words $w_0, \dots, w_{t_i} \in \mathbb{N}^*$ together with numbers $t_0 < t_1 < \dots < t_i$ and bits $b_0, \dots, b_{i-1} \in \{0, 1\}$ we describe how we determine $t_{i+1} \in \mathbb{N}$, $w_{t_i+1}, \dots, w_{t_{i+1}} \in \mathbb{N}^*$ and $b_i \in \{0, 1\}$ in stage i of the construction.³ We let $r_0 := p|_0 w_0 \hat{0} = \hat{0}$ and $r_{n+1} := (p|_{n+1} w_{t_{n+1}} \hat{0}, \dots, p|_{n+1} w_{t_{n+1}} \hat{0})$ for all $n < i$. Now we consider the question whether the i -th Turing machine halts on

$$\langle r_0, \dots, r_i, p|_{i+1} q_{t_i+1}, p|_{i+1} q_{t_i+2}, p|_{i+1} q_{t_i+3}, \dots \rangle$$

³For notational clarity, we have underlined the indexes of t ; the line has no mathematical meaning.

for some $q_{t_i+n} \in \mathbb{N}^{\mathbb{N}}$ with $n \in \mathbb{N}$. Whether or not this is the case can be decided with the help of $0'$ since the answer depends only on the finite portion $p|_{i+1}$ of p read so far. If the outcome of the decision is positive, then already a finite number $q_{t_i+1}, \dots, q_{t_{i+1}} \in \mathbb{N}^{\mathbb{N}}$ for some $t_{i+1} > t_i$ and, in fact, finite prefixes of these are sufficient for the machine i to halt. In this case we can compute such a t_{i+1} and corresponding words $w_{t_i+n} \in \mathbb{N}^*$ such that $q_j := w_j \widehat{0}$ for $j \in \{t_i + 1, \dots, t_{i+1}\}$ satisfy the condition, and we set $b_i := 1$. Otherwise, if the outcome of the decision is negative, then we choose $w_{t_{i+1}} := \varepsilon$, $t_{i+1} := t_i + 1$ and $b_i := 0$. If we continue inductively in this way, then we can compute $I(p) := \langle r_0, r_1, r_2, \dots \rangle$ with the help of $0'$ uniformly in p . Since we have used longer and longer prefixes $p|_n$ of p in the construction of the r_n , the value $I(p)$ satisfies $\lim \circ I(p) = p$. Moreover, the construction guarantees that we can compute $J \circ I$ with the help of $0'$, namely $J \circ I(p) = (b_0, b_1, b_2, \dots)$. \square

We note that H is, in particular, continuous, and $H^{-1} = \lim \circ J^{-1}$ is computable by Theorem 2.14. Theorem 3.1 yields the following characterization of functions computable relative to the halting problem. The interesting point of the following theorem is not that there is an object H that characterizes functions that are computable relative to the halting problem (the problem $\langle \text{id} \times \chi_{0'} \rangle$ with the characteristic function $\chi_{0'}$ of the halting problem would be a simpler such example), but the point is that our particular H has this property.

Theorem 3.2 (Computability relative to the halting problem). *A function $F : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ is computable relative to the halting problem if and only if there is a computable function $G : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ such that $F = G \circ H$.*

Proof. By Theorem 3.1 H is computable relative to $0'$. Hence $G \circ H$ is computable relative to the halting problem for every computable $G : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$. Let now F be computable relative to the halting problem $0'$. Then there is a computable $S : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$, such that $F(p) = S\langle 0', p \rangle$ for all $p \in \text{dom}(F)$. By Theorem 2.14 we know that there is a computable function $T : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ such that $T \circ J = \text{lim}$, i.e., $T \circ H = \text{id}$. On the other hand, there is a computable function $r : \mathbb{N} \rightarrow \mathbb{N}$ such that Turing machine $r(i)$ halts on all inputs if and only if machine i halts on input $\widehat{0}$. Hence $J(\widehat{0})(i) = J(p)(r(i))$ for all $p \in \mathbb{N}^{\mathbb{N}}$ and $i \in \mathbb{N}$. Then $R : \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ with $R(q)(i) := qr(i)$ is computable and so is $G := S \circ \langle R, T \rangle$. We obtain

$$F(p) = S\langle J(\widehat{0}), p \rangle = S\langle R \circ H(p), T \circ H(p) \rangle = G \circ H(p)$$

for all $p \in \text{dom}(F)$, hence $F = G \circ H$ if G is restricted suitably (which is possible since $H(p) \cap H(q) = \emptyset$ for $p \neq q$.) \square

We note that this result implies that H is not computable (hence it is an interesting example of a natural problem that is continuous and not computable). Another interesting consequence of our results is that the functions that are computable relative to the halting problem form the largest class of functions that, when composed with a limit computable function from the right, yield a limit computable function.

Corollary 3.3 (Computability relative to the halting problem). *A function $F : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ is computable relative to the halting problem if and only if $F \circ G$ is limit computable for every limit computable $G : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$.*

The “if” direction follows from Theorems 3.2 and 2.14, when one chooses $G = \text{lim}$. The “only if” direction is a consequence of Theorems 3.2 and 2.6. A comparison of Corollaries 3.3 and 2.28 shows that the notions of a low function is dual to the notion of a function that

is computable relative to the halting problem. Both classes should be naturally studied alongside the limit computable functions.

As another non-uniform side result of the limit inversion theorem (Theorem 3.1) we obtain a classical result from computability theory, the Friedberg jump inversion theorem [21].

Corollary 3.4 (Friedberg jump inversion theorem 1957). *For every $q \in \mathbb{N}^{\mathbb{N}}$ with $0' \leq_T q$ there exists a $p \in \mathbb{N}^{\mathbb{N}}$ with $p' \equiv_T q$.*

Proof. Given a fixed $q \in \mathbb{N}^{\mathbb{N}}$ with $0' \leq_T q$ we can compute $p := I(q)$ and $p' = J \circ I(q)$ with I from the proof of Theorem 3.1. In particular, $p' \leq_T q$. On the other hand, by the same theorem and Theorem 2.14 we obtain $q = \lim(p) \leq_T p'$. \square

As another corollary we obtain the announced extension of Corollary 2.8.

Corollary 3.5. *For every function $F : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ that is computable relative to the halting problem there exists a computable function $G : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ such that $\lim \circ G = F \circ \lim$.*

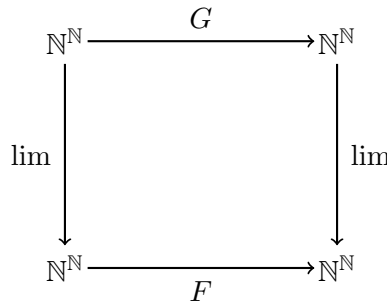


Figure 1: Limit diagram.

This result can be interpreted as a statement about the diagram in Figure 1. Vice versa, if G is an arbitrary computable function that transfers converging sequences into converging sequences (not necessarily in an extensional way), then we can mimic the behavior of G on the limits by a function that is computable relative to the halting problem. This follows from Theorems 3.2 and 2.14 and was proved directly with a somewhat more involved proof that does not exploit the Galois connection between Turing jumps and limits in [11, Theorem 14.11].

Corollary 3.6 (B., Hendtlass and Kreuzer 2017). *For all computable $G : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ the multi-valued function $\lim \circ G \circ \lim^{-1}$ is computable relative to the halting problem.*

We close this section with a uniform version of Corollary 3.6. For the formulation we need a total representations Φ of all continuous functions $F : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ with computable G_δ -domain that satisfies a utm- and smn-theorem [52, Theorem 2.3.5]. The result is tailor-made to yield a proof of Theorem 4.12 (3).

Theorem 3.7 (Uniform limit control theorem). *There exists a computable $R : \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ such that $\Phi_{\lim \circ R(q)}(p) \in \lim \circ \Phi_q \circ \lim^{-1}(p)$ for all $p, q \in \mathbb{N}^{\mathbb{N}}$ with $p \in \text{dom}(\lim \circ \Phi_q \circ \lim^{-1})$.*

Proof. By a relativized version of Theorem 3.1 there exist computable $I, S : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ such that $\lim \circ I_q = \text{id}$ and $S_{q'} = J_q \circ I_q$ for every $q \in \mathbb{N}^{\mathbb{N}}$. This can be proved exactly as in

the proof of Theorem 3.1 except that we check whether the i -th Turing machine halts upon input of

$$\langle q, \langle r_0, \dots, r_i, p|_{i+1}q_{t_i+1}, p|_{i+1}q_{t_i+2}, p|_{i+1}q_{t_i+3}, \dots \rangle \rangle,$$

which can be decided using q' . Now, by applying the jump normal form theorem (Theorem 2.14) to Φ we obtain a computable function H with $\lim \circ \Phi_q = H \circ J_q$. Then $H \circ S$ is computable, and by the smn-theorem for Φ there exists a total computable function P such that $\Phi_{P(r)}(p) = H \circ S \langle r, p \rangle$ for all $r, p \in \mathbb{N}^{\mathbb{N}}$. Finally, by the limit normal form theorem (Theorem 2.6) there exists a computable function R such that $\lim \circ R = P \circ J$. Such an R is necessarily total. Altogether, we obtain

$$\Phi_{\lim R(q)}(p) = \Phi_{PJ(q)}(p) = HS \langle q', p \rangle = HJ_q I_{q'}(p) = \lim \Phi_q I_{q'}(p) \in \lim \circ \Phi_q(\lim^{-1}(p))$$

for all $p, q \in \mathbb{N}^{\mathbb{N}}$ with $\lim^{-1}(p) \subseteq \text{dom}(\lim \circ \Phi_q)$. \square

We note that the basic problems \lim, J^{-1} are generators of a monoid which can be used to characterize all sorts of other computability theoretic properties. For instance $J^{-1}J^{-1}\lim \lim$ characterizes the property of being *low*₂ and $J^{-1}\lim \lim$ the property of being *low relative to the halting problem* (see also [9]).

4. LIMIT COMPUTABILITY ON REPRESENTED SPACES

The purpose of this section is to transfer our results on limit computable maps to represented spaces. We recall that a *representation* of a set X is a surjective function $\delta : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow X$. In this situation we say that (X, δ) is a *represented space*. Any point $p \in \mathbb{N}^{\mathbb{N}}$ with $\delta(p) = x$ is called a *name* of x , and the word “representation” is reserved for the map δ as such. A *problem* is a multi-valued partial function $f : \subseteq X \rightrightarrows Y$ on represented spaces X and Y . If (X, δ_X) and (Y, δ_Y) are represented spaces and $f : \subseteq X \rightrightarrows Y$ is a problem, then $F : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ *realizes* f , in symbols $F \vdash f$, if $\delta_Y F(p) \in f \delta_X(p)$ holds for all $p \in \text{dom}(f \delta_X)$.

Definition 4.1 (Computable functions). A problem f is called *continuous*, *computable*, *limit computable*, *low* or *computable with respect to the halting problem*, if it has a realizer with the corresponding property.

Likewise, other properties can be transferred from realizers to problems. By $\mathcal{C}(X, Y)$ we denote the *set of total continuous functions* $f : X \rightarrow Y$. It is clear that all properties related to closure under composition that have been discussed in the previous section can be transferred to problems. This is because $F \vdash f$ and $G \vdash g$ implies $F \circ G \vdash f \circ g$. We can also transfer properties of points to represented spaces using representations.

Definition 4.2 (Computable points). A point $x \in X$ in a represented space X is called *computable*, *limit computable* or *low* if it has a name with the corresponding property.

We note that the notion of 1-genericity is an example of a property that should not be defined via names since it is not invariant under equivalent representations (see Proposition 5.23 and the discussion afterwards).

Our main goal here is to transfer the limit normal form theorem and the jump normal form theorem (Theorems 2.6 and 2.14) to problems on represented spaces. Since we can incorporate limits and jumps into the represented spaces, these normal forms can be expressed very neatly. For this purpose we need the following concepts of jumps on represented spaces.

Definition 4.3 (Jumps). Let (X, δ_X) be a represented space and $\delta := \delta_X$, let $T : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ be surjective, and let $S : \subseteq \mathbb{N}^{\mathbb{N}} \rightrightarrows \mathbb{N}^{\mathbb{N}}$ be such that $S^{-1} : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ is single-valued and surjective. We define the following representations of X :

- (1) $\delta^T := \delta_{X^T} := \delta_X \circ T$ (T -jump)
- (2) $\delta_S := \delta_{X_S} := \delta_X \circ S^{-1}$ (S^{-1} -jump)

We denote the represented spaces (X, δ_{X^T}) and (X, δ_{X_S}) for short by X^T and X_S , respectively. In the special case of $T = \lim$ and $T = \lim_{\Delta}$, we define

- (3) $\delta' := \delta_{X'} := \delta_X \circ \lim$ (jump)
- (4) $\delta^{\Delta} := \delta_{X^{\Delta}} := \delta_X \circ \lim_{\Delta}$ (discrete jump)

We denote the represented spaces $(X, \delta_{X'})$ and $(X, \delta_{X^{\Delta}})$ also by X' and X^{Δ} , respectively.

We will apply this concept of a jump in the case of $T = \lim$, $T = L$, $S = H$ and $S = J$. The special jumps for $T = \lim$ and $T = \lim_{\Delta}$ (the limit on Baire space with respect to the discrete metric) were originally defined by Ziegler [56] and later also studied by the author, de Brecht and Pauly [7]. More general concepts of jumps were also studied in the context of effective descriptive set theory by de Brecht [15] and de Brecht and Pauly [38, 39, 40].

A common feature of the maps $\lim, \lim_{\Delta}, J^{-1}, L$ and H^{-1} is that they are surjective and transparent. It was noted by de Brecht [15] that the functor $X \mapsto X^T$ for surjective and transparent T can be seen as an endofunctor on the class of represented spaces (that leaves the maps unchanged). This is made precise by the following result.

Proposition 4.4 (The jump as endofunctor). *If $f : \subseteq X \rightrightarrows Y$ is a computable problem and $T : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ is transparent and surjective then it follows that f considered as a problem of type $f : \subseteq X^T \rightrightarrows Y^T$ is computable too.*

Proof. If $f : \subseteq X \rightrightarrows Y$ is computable, then it has a computable realizer $F : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$. Since T is transparent, there is a computable $G : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ with $F \circ T = T \circ G$. Hence G is a computable realizer of $f : \subseteq X^T \rightrightarrows Y^T$. \square

Through some results presented in this section we will get a better understanding of the types of problems mentioned in Proposition 4.4 for the specific maps T that we are interested in.

We start with a result that characterizes limit computable problems and shows that the jump Y' on the output side can be balanced by X_J on the input side.

Theorem 4.5 (Limit computability). *Let $f : \subseteq X \rightrightarrows Y$ be a problem. Then the following are equivalent:*

- (1) $f : \subseteq X \rightrightarrows Y$ is limit computable,
- (2) $f : \subseteq X \rightrightarrows Y'$ is computable,
- (3) $f : \subseteq X_J \rightrightarrows Y$ is computable.

Proof. The equivalences are consequences of the limit normal form theorem (Theorem 2.6) and the jump normal form theorem (Theorem 2.14). \square

We note that this result can be interpreted such that jumps X' and X_J are adjoint to each other. However, this is a pure computability theoretic adjointness that does not relativize to a topological adjointness. The proof of the direction (3) \implies (2) can be extended to a uniform proof, but this is not so for the direction (2) \implies (3). In fact, $\mathcal{C}((\mathbb{N}^{\mathbb{N}})_J, \mathbb{N}^{\mathbb{N}}) \not\subseteq \mathcal{C}(\mathbb{N}^{\mathbb{N}}, (\mathbb{N}^{\mathbb{N}})')$, since J^{-1} is not topologically transparent. However, we can derive the following relativized version of Theorem 4.5 from Theorem 2.18.

Corollary 4.6 (Relativized limit computability). *Let $f : \subseteq X \rightrightarrows Y$ be a problem and $q \in \mathbb{N}^{\mathbb{N}}$. Then the following are equivalent:*

- (1) $f : \subseteq X \rightrightarrows Y$ is limit computable relative to q ,
- (2) $f : \subseteq X \rightrightarrows Y'$ is computable relative to q ,
- (3) $f : \subseteq X_{J_q} \rightrightarrows Y$ is computable.

We can transfer our characterization of low problems.

Theorem 4.7 (Low computability). *Let $f : \subseteq X \rightrightarrows Y$ be a problem. Then the following are equivalent:*

- (1) $f : \subseteq X \rightrightarrows Y$ is low,
- (2) $f : \subseteq X \rightrightarrows Y^{\text{L}}$ is computable,
- (3) $f : \subseteq X^{\text{L}} \rightrightarrows Y^{\text{L}}$ is computable,
- (4) $f : \subseteq X_{\text{J}} \rightrightarrows Y_{\text{J}}$ is computable,
- (5) $f : \subseteq X_{\text{J}} \rightrightarrows Y_{\text{J}}$ is computable relative to the halting problem $0'$.

Proof. The equivalence of (1) and (2) is a direct consequence of the definition of low maps on Baire space. Since low maps are closed under composition by Proposition 2.27, it follows that (2) implies (3). Since $\text{id} : X \rightarrow X^{\text{L}}$ is computable, it follows that (3) implies (2). By Theorem 4.5 (2) and (4) are equivalent since $X^{\text{L}} = (X_{\text{J}})'$. By the statement “(1) \iff (4)” of Theorem 4.8 (which independently follows from Corollaries 3.5 and 3.6) we obtain that (3) and (5) are equivalent. \square

Finally, we obtain a dual characterization of the problems that are computable relative to the halting problem.⁴

Theorem 4.8 (Computability relative to the halting problem). *Let $f : \subseteq X \rightrightarrows Y$ be a problem. Then the following are equivalent:*

- (1) $f : \subseteq X \rightrightarrows Y$ is computable relative to the halting problem $0'$,
- (2) $f : \subseteq X_{\text{H}} \rightrightarrows Y$ is computable,
- (3) $f : \subseteq X_{\text{H}} \rightrightarrows Y_{\text{H}}$ is computable,
- (4) $f : \subseteq X' \rightrightarrows Y'$ is computable,
- (5) $f : \subseteq X' \rightrightarrows Y'$ is low.

Proof. The equivalence of (1) and (2) is a consequence of Theorem 3.2. Since functions computable relative to $0'$ are closed under composition, it follows again by Theorem 3.2 that (2) implies (3). Since $\text{id} : Y_{\text{H}} \rightarrow Y$ is computable, it is clear that (3) implies (2). The equivalence of (2) and (4) follows from Theorem 4.5 since $X_{\text{H}} = (X')_{\text{J}}$. The equivalence of (3) and (5) follows from Theorem 4.7. \square

The equivalence of (1) and (4) can also be directly derived from Corollaries 3.5 and 3.6. Likewise we obtain the following relativized version.

Theorem 4.9 (Computability relative to an oracle). *Let $f : \subseteq X \rightrightarrows Y$ be a problem and $q \in \mathbb{N}^{\mathbb{N}}$. Then the following are equivalent:*

- (1) $f : \subseteq X \rightrightarrows Y$ is computable relative to q' ,
- (2) $f : \subseteq X' \rightrightarrows Y'$ is computable relative to q .

⁴We note that the equivalence of (1) and (4) shows that [55, Theorem 23 (c)] is not correct.

Proof. The implication from (2) to (1) is a consequence of Theorem 3.7. For the implication from (1) to (2) we need a relativized version of Corollary 3.5 that can actually be derived from this corollary. Let $U\langle q, p \rangle := \Phi_q(p)$ be the universal computable function. Then by Corollary 3.5 there exists a computable G such that $\lim \circ G = U \circ \lim$. There is also a computable $R : \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ with $\lim \circ R\langle q, p \rangle = \langle \lim q, \lim p \rangle$. If we denote by R_q the function with $R_q(p) := R\langle q, p \rangle$ then we obtain $\lim \circ G \circ R_q = \Phi_{\lim q} \circ \lim$. Hence, $G \circ R_q$ is a realizer for $f : \subseteq X' \rightrightarrows Y'$ if $\Phi_{\lim q}$ is a realizer for $f : \subseteq X \rightrightarrows Y$. This proves the claim. \square

We immediately obtain the following corollary, where $q^{(n)}$ denotes the n -th Turing jump of $q \in \mathbb{N}^{\mathbb{N}}$ and $X^{(n)}$ denotes the n -th jump of the represented space X .

Corollary 4.10 (Computability relative to higher jumps). *Let $f : \subseteq X \rightrightarrows Y$ be a problem and $n \in \mathbb{N}$. Then the following are equivalent:*

- (1) $f : \subseteq X \rightrightarrows Y$ is computable relative to $0^{(n)}$,
- (2) $f : \subseteq X^{(n)} \rightrightarrows Y^{(n)}$ is computable.

Other characterizations can be derived as conclusions of the results provided in this section. Now we want to show that jumps interact nicely with products and function space constructions. We recall that for two represented spaces (X, δ_X) and (Y, δ_Y) we can define a representation of $X \times Y$, of $X^{\mathbb{N}}$ and of $\mathcal{C}(X, Y)$ as follows:

- (1) $\delta_{X \times Y} : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow X \times Y, \langle p, q \rangle \mapsto (\delta_X(p), \delta_Y(q))$.
- (2) $\delta_{X^{\mathbb{N}}} : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow X^{\mathbb{N}}, \langle p_0, p_1, p_2, \dots \rangle \mapsto (\delta_X(p_n))_{n \in \mathbb{N}}$.
- (3) $\delta_{\mathcal{C}(X, Y)} : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathcal{C}(X, Y)$ by $\delta_{\mathcal{C}(X, Y)}(p) = f : \iff \Phi_p \vdash f$.

It is known that these representations make evaluation and currying computable [52, Lemmas 3.3.14, 3.3.16 and Theorem 3.3.15] and that sequences can be identified with continuous functions.

Fact 4.11 (Function space). *The following are computable for all represented spaces X, Y :*

- (1) $\text{ev} : \mathcal{C}(X, Y) \times X \rightarrow Y, (f, x) \mapsto f(x)$,
- (2) $\text{cur} : \mathcal{C}(X \times Y, Z) \rightarrow \mathcal{C}(X, \mathcal{C}(Y, Z)), f \mapsto (x \mapsto (y \mapsto f(x, y)))$,
- (3) $\text{id} : \mathcal{C}(\mathbb{N}, X) \rightarrow X^{\mathbb{N}}$ and its inverse.

We obtain the following result that shows how jumps interact with products and function space constructions. We call a problem $f : X \rightarrow Y$ a *computable isomorphism* if f is computable and bijective and its inverse f^{-1} is computable too.

Theorem 4.12 (Jumps with products and exponentials). *Let X and Y be represented spaces. Then the following are computable isomorphisms:*

- (1) $\text{id} : (X \times Y)' \rightarrow X' \times Y'$,
- (2) $\text{id} : (X^{\mathbb{N}})' \rightarrow (X')^{\mathbb{N}}$,
- (3) $\text{id} : \mathcal{C}(X, Y)' \rightarrow \mathcal{C}(X', Y')$.

In particular, $\mathcal{C}(X', Y')$ is exactly the set of continuous functions $f : X \rightarrow Y$.

Proof. The first two statements follow from the fact that the tupling functions are continuous:

$$\lim_{i \rightarrow \infty} \langle p_i, q_i \rangle = \langle \lim_{i \rightarrow \infty} p_i, \lim_{i \rightarrow \infty} q_i \rangle \text{ and } \lim_{i \rightarrow \infty} \langle p_{0i}, p_{1i}, \dots \rangle = \langle \lim_{i \rightarrow \infty} p_{0i}, \lim_{i \rightarrow \infty} p_{1i}, \dots \rangle.$$

Now we consider the evaluation map $\text{ev} : \mathcal{C}(X, Y) \times X \rightarrow Y$, which is computable. By Proposition 4.4 it follows that also $\text{ev} : (\mathcal{C}(X, Y) \times X)' \rightarrow Y'$ is computable, and hence by (1) we obtain that $\text{ev} : \mathcal{C}(X, Y)' \times X' \rightarrow Y'$ is computable. Since currying is computable,

it follows that $\text{id} : \mathcal{C}(X, Y)' \rightarrow \mathcal{C}(X', Y')$ is computable. We still need to prove that $\text{id} : \mathcal{C}(X', Y') \rightarrow \mathcal{C}(X, Y)'$ is computable. But this follows from the uniform version of the uniform limit control theorem (Theorem 3.7). \square

We should warn the reader that we have an ambiguity in the terminology that one has to keep in mind and that is expressed in the following corollary.

Corollary 4.13 (Limit computable points in function spaces). *Let X, Y be represented spaces and $f \in \mathcal{C}(X, Y)$. Then f is limit computable as a point in $\mathcal{C}(X, Y)$ if and only if $f : X \rightarrow Y$ is computable relative to the halting problem as a function.*

However, one can also see this as fit in terminology since for points in Baire space being limit computability is equivalent to being computable relative to the halting problem by Shoenfield's limit lemma (Corollary 2.17). Hence f being computable relative to the halting problem as a point in $\mathcal{C}(X, Y)$ is equivalent to $f : X \rightarrow Y$ being computable relative to the halting problem as a function. The equivalence between computability relative to the halting problem and limit computability for points extends to sequences. Theorem 4.12 together with the fact that $\text{id} : \mathcal{C}(\mathbb{N}, X) \rightarrow X^{\mathbb{N}}$ is a computable isomorphism imply the following.

Corollary 4.14 (Limit computable sequences). *The identity $\text{id} : \mathcal{C}(\mathbb{N}, X)' \rightarrow \mathcal{C}(\mathbb{N}, X')$ is a computable isomorphism. In particular, the limit computable functions $f : \mathbb{N} \rightarrow X$ are exactly the functions $f : \mathbb{N} \rightarrow X$ that are computable relative to the halting problem.*

The jump $X \mapsto X_J$ does not preserve products and function spaces in the way $X \mapsto X'$ does according to Theorem 4.12. Basically, all our negative results in this direction can be derived from Spector's jump inversion theorem (see [48] or [36, Proposition V.2.26]). In particular, it implies that the notion of a low function is incomparable with the notion of a function that is computable relative to a low oracle.⁵

Proposition 4.15 (Low functions). *There is a Lipschitz continuous function $F : \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ that is computable relative to a low oracle, but that is not low as a function. There is also a low function $f : \{0, 1\} \rightarrow \mathbb{N}^{\mathbb{N}}$ that is not computable relative to a low oracle.*

Proof. By Spector's jump inversion theorem there are low $p, q \in \mathbb{N}^{\mathbb{N}}$ such that $\langle p, q \rangle$ is not low. Hence the function $F : \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}, r \mapsto \langle q, r \rangle$ does not map the low $p \in \mathbb{N}^{\mathbb{N}}$ to a low value $F(p) = \langle q, p \rangle$. That is, F is not low, but computable relative to the low point q . It is easy to see that F is Lipschitz continuous with Lipschitz constant 1.

There are also computable $r, s \in \mathbb{N}^{\mathbb{N}}$ such that $L(r) = p$ and $L(s) = q$. We consider the functions $f, g : \{0, 1\} \rightarrow \mathbb{N}^{\mathbb{N}}$ defined by

$$f(i) := \begin{cases} p & \text{if } i = 0 \\ q & \text{otherwise} \end{cases} \quad \text{and} \quad g(i) := \begin{cases} r & \text{if } i = 0 \\ s & \text{otherwise} \end{cases} .$$

Then g is computable and $f = L \circ g$. Hence f is low. Let us assume that f is computable with respect to a low oracle $t \in \mathbb{N}^{\mathbb{N}}$. Then $p = f(0) \leq_T t$ and $q = f(1) \leq_T t$ follows and hence $\langle p, q \rangle \leq_T t$, which is a contradiction since $\langle p, q \rangle$ is not low. \square

We will see several further low functions that are not computable relative to a low oracle in section 6.

⁵A referee provided an interesting alternative proof of this result, using Chaitin's $\Omega = \langle \Omega_0, \Omega_1 \rangle$. The even and odd parts Ω_0 and Ω_1 , respectively, are low according to [16, Theorem 15.2.3] and hence $F : p \mapsto \langle \Omega_0, p \rangle$ and $f : i \mapsto \Omega_i$ are alternative concrete examples that satisfy Proposition 4.15.

As a preparation for the following results we show that relative computability can also be characterized via function spaces if the corresponding oracle class is closed downwards by Turing reducibility.

Lemma 4.16 (Relative computability). *Let $F : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ be a function and $q \in \mathbb{N}^{\mathbb{N}}$. Then the following are equivalent:*

- (1) F is computable relative to q ,
- (2) $(\exists r \leq_T q)(\forall p \in \text{dom}(F)) F(p) = \Phi_r(p)$.

Proof. (1) implies (2) by the smn-theorem and (2) implies (1) by the utm-theorem. \square

We can apply this in particular to the class of functions that are computable with respect to a low oracle.

Corollary 4.17 (Computability relative to a low oracle). *Let X, Y be represented spaces. Then $\mathcal{C}(X, Y)^{\text{L}}$ is the class of functions that are computable relative to a low oracle.*

Now we get the following negative result on how L and J behave on function spaces. We note that the given identities are not surjective, since the spaces $\mathcal{C}(X^{\text{L}}, Y^{\text{L}})$ and $\mathcal{C}(X_{\text{J}}, Y_{\text{J}})$ contain discontinuous functions.

Proposition 4.18 (Function spaces for low functions). *For instance for $X = Y = \mathbb{N}^{\mathbb{N}}$ the embeddings $\text{id} : \mathcal{C}(X, Y)^{\text{L}} \rightarrow \mathcal{C}(X^{\text{L}}, Y^{\text{L}})$ and $\text{id} : \mathcal{C}(X, Y)_{\text{J}} \rightarrow \mathcal{C}(X_{\text{J}}, Y_{\text{J}})$ are not computable. The partial inverses of these maps are also not computable.*

Proof. The function F from Proposition 4.15 is not a computable point in $\mathcal{C}(X^{\text{L}}, Y^{\text{L}})$, but by Corollary 4.17 it is a computable point in $\mathcal{C}(X, Y)^{\text{L}}$. This means that the embedding $\text{id} : \mathcal{C}(X, Y)^{\text{L}} \rightarrow \mathcal{C}(X^{\text{L}}, Y^{\text{L}})$ is not computable. With Theorem 4.12 it follows that also the embedding $\text{id} : \mathcal{C}(X, Y)_{\text{J}} \rightarrow \mathcal{C}(X_{\text{J}}, Y_{\text{J}})$ is not computable since $Z^{\text{L}} = (Z_{\text{J}})'$. The function f from Proposition 4.15 can easily be converted into a function $f : \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ with corresponding properties, and hence $\mathcal{C}(X^{\text{L}}, Y^{\text{L}})$ contains computable points which are not computable in $\mathcal{C}(X, Y)^{\text{L}}$. The space $\mathcal{C}(X, Y)_{\text{J}}$ does not even contain any computable points whatsoever. Altogether, this shows that the partial inverses of the given maps are not computable. \square

The positive properties that apply to Turing jumps and products are captured in the following result.

Proposition 4.19 (Turing jumps and products). *Let X and Y be represented spaces. Then the following maps are computable:*

- (1) $\text{id} : (X \times Y)_{\text{J}} \rightarrow X_{\text{J}} \times Y_{\text{J}}$,
- (2) $\text{id} : (X^{\mathbb{N}})_{\text{J}} \rightarrow (X_{\text{J}})^{\mathbb{N}}$.

In general, these maps are not computable isomorphisms.

Proof. It is easy to see that $\langle \text{J} \times \text{J} \rangle$ is limit computable since J is limit computable. Hence, by the jump normal form theorem (Theorem 2.14) there exists a computable F such that $F\text{J}\langle p, q \rangle = \langle \text{J}(p), \text{J}(q) \rangle$. This function F is a realizer of $\text{id} : (X \times Y)_{\text{J}} \rightarrow X_{\text{J}} \times Y_{\text{J}}$. Likewise, limit computability of $\langle \widehat{\text{J}} \rangle$ implies that $\text{id} : (X^{\mathbb{N}})_{\text{J}} \rightarrow (X_{\text{J}})^{\mathbb{N}}$ is computable. Let us assume that the inverse of $\text{id} : (X \times Y)_{\text{J}} \rightarrow X_{\text{J}} \times Y_{\text{J}}$ is computable. Then by Proposition 4.4 and Fact 4.11 the evaluation $\text{ev} : \mathcal{C}(X, Y)_{\text{J}} \times X_{\text{J}} \rightarrow Y_{\text{J}}$ would be computable, and hence $\text{id} : \mathcal{C}(X, Y)_{\text{J}} \rightarrow \mathcal{C}(X_{\text{J}}, Y_{\text{J}})$ would be computable, which is not the case in general by Proposition 4.18, for instance for $X = Y = \mathbb{N}^{\mathbb{N}}$. If the inverse of $\text{id} : (X \times Y)_{\text{J}} \rightarrow X_{\text{J}} \times Y_{\text{J}}$

for binary products of $X = Y = \mathbb{N}^{\mathbb{N}}$ is not computable, it follows that the inverse of $\text{id} : (X^{\mathbb{N}})_{\text{J}} \rightarrow (X_{\text{J}})^{\mathbb{N}}$ for countable products of $X = \mathbb{N}^{\mathbb{N}}$ is not computable either. \square

We mention that the jump $X \mapsto X_{\text{J}}$ commutes with coproducts while the jump $X \mapsto X'$ does not. However, we do not need these facts here. Hence, we leave the details to the reader. Here we just formulate a corollary for $X \mapsto X^{\text{L}}$ that we obtain from Proposition 4.19 and Theorem 4.12.

Corollary 4.20 (Lowness and products). *Let X and Y be represented spaces. Then the following maps are computable:*

- (1) $\text{id} : (X \times Y)^{\text{L}} \rightarrow X^{\text{L}} \times Y^{\text{L}}$,
- (2) $\text{id} : (X^{\mathbb{N}})^{\text{L}} \rightarrow (X^{\text{L}})^{\mathbb{N}}$.

In general, these maps are not computable isomorphisms.

That the maps are not computable isomorphisms follows again from Spector's jump inversion theorem. As a consequence of Corollary 4.20 and Fact 4.11 we obtain at least one positive implication for sequences that are computable relative to a low oracle. The inverse implication is not correct in general by Proposition 4.15.

Corollary 4.21 (Low sequences). *The identity $\text{id} : \mathcal{C}(\mathbb{N}, X)^{\text{L}} \rightarrow \mathcal{C}(\mathbb{N}, X^{\text{L}})$ is computable. In particular, every function $f : \mathbb{N} \rightarrow X$ that is computable relative to a low oracle is also low as a function.*

In the remainder of this section we briefly want to discuss some interpretations of our results in the lattice of representations. We recall that given two represented spaces (X, δ_X) and (Y, δ_Y) with $X \subseteq Y$ we say that δ_X is *reducible* to δ_Y , in symbols $\delta_X \leq \delta_Y$, if $\text{id} : (X, \delta_X) \rightarrow (Y, \delta_Y)$ is computable. The corresponding equivalence is denoted by \equiv .

As an immediate corollary of Proposition 4.4 we obtain the following.

Corollary 4.22 (Monotonicity of jumps). *Let (X, δ_X) and (Y, δ_Y) be represented spaces and let $T : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ be transparent and surjective. Then $\delta_X \leq \delta_Y$ implies $\delta_X^T \leq \delta_Y^T$.*

We now consider the jump operations that we have introduced on represented spaces. Firstly, they are all ordered in the following way.

Corollary 4.23 (Order of jumps). *For every representation δ each of the following reductions holds: $\delta_{\text{J}} \leq \delta_{\text{H}} \leq \delta \leq \delta^{\Delta} \leq \delta^{\text{L}} \leq \delta'$.*

Here the first reduction holds since lim has a computable right inverse $F : \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$, and hence $F : (\mathbb{N}^{\mathbb{N}})_{\text{J}} \rightarrow (\mathbb{N}^{\mathbb{N}})_{\text{J}}$ is computable by Proposition 4.4; the second reduction holds since H^{-1} is computable; the third reduction holds since lim_{Δ} has a computable right inverse; the fourth reduction holds in lim_{Δ} is low by Corollary 2.29 and Theorem 2.30, and the last reduction holds since L is limit computable by Proposition 2.27. In fact, some of the discussed operations on represented spaces have further properties. In particular the pair $(\delta \mapsto \delta_{\text{J}}, \delta \mapsto \delta')$ forms a *Galois connection* in the lattice of representations of a fixed set. This is a consequence of Theorem 4.5 applied to the injection $\text{id} : X \rightarrow Y$.

Corollary 4.24 (Galois connection between Turing jumps and limits). *Let (X, δ_X) and (Y, δ_Y) be represented spaces with $X \subseteq Y$. Then $\delta_{X_{\text{J}}} \leq \delta_Y \iff \delta_X \leq \delta_{Y'}$.*

The operation $\delta \mapsto \delta^{\text{L}} = (\delta_{\text{J}})'$ is the *monad* of this Galois connection, which implies that it is a closure operator. Likewise, $\delta \mapsto \delta_{\text{H}} = (\delta')_{\text{J}}$ is an interior operator [17, Proposition 3(4)].

Proposition 4.25 (Closure and interior operators). $\delta \mapsto \delta^{\text{L}}$ and $\delta \mapsto \delta^{\Delta}$ are closure operators and $\delta \mapsto \delta_{\text{H}}$ is an interior operator on the lattice of representations of a fixed set X .

The fact that $\delta \mapsto \delta^{\Delta}$ is also a closure operator can easily be proved directly. Another consequence of Corollary 4.24 which can also easily be proved directly, is that $\delta \mapsto \delta_{\text{J}}$ preserves suprema and $\delta \mapsto \delta'$ preserves infima [17, Proposition 3(8)]. We recall for represented spaces (X, δ_X) and (Y, δ_Y) the *meet* or *infimum* $\delta_X \wedge \delta_Y$, which is a representation of $X \cap Y$, can be defined by $(\delta_X \wedge \delta_Y)\langle p, q \rangle = z : \iff \delta_X(p) = \delta_Y(q) = z$. If $X = Y$, then $\delta_X \wedge \delta_Y$ is actually known to be the infimum in the lattice of representations of X [52, Lemma 3.3.8]. Hence we obtain the following.

Proposition 4.26 (Infimum and jumps). *Let (X, δ_X) and (Y, δ_Y) be represented spaces. Then $(\delta_X \wedge \delta_Y)' \equiv \delta'_X \wedge \delta'_Y$.*

Analogously, $\delta \mapsto \delta'$ commutes with products (see Theorem 4.12), which also implies Proposition 4.26 since $\delta_X \wedge \delta_Y = \Delta_{X \cap Y}^{-1} \circ \delta_{X \times Y}$, where $\Delta_{X \cap Y}$ denotes the *diagonal* $\Delta_{X \cap Y} : X \cap Y \rightarrow (X \cap Y) \times (X \cap Y), x \mapsto (x, x)$. Likewise, $\delta \mapsto \delta_{\text{J}}$ commutes with coproducts and suprema, but we do not formulate these results here.

5. LIMIT COMPUTABILITY ON METRIC SPACES

In this section we want to transfer some of our results to computable metric spaces. We recall that (X, d, α) is called a *computable metric space*, if (X, d) is a metric space with metric $d : X \times X \rightarrow \mathbb{R}$ and $\alpha : \mathbb{N} \rightarrow X$ is a sequence that is dense in X such that $d \circ (\alpha \times \alpha) : \mathbb{N}^2 \rightarrow \mathbb{R}$ is a computable sequence of real numbers. Each computable metric space is equipped with its *Cauchy representation* $\delta_X : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow X$ that is defined by $\delta_X(p) := \lim_{n \rightarrow \infty} \alpha p(n)$ with $\text{dom}(\delta_X) := \{p \in \mathbb{N}^{\mathbb{N}} : (\forall k)(\forall n \geq k) d(\alpha p(n), \alpha p(k)) < 2^{-k} \text{ and } (\alpha p(n))_{n \in \mathbb{N}} \text{ converges}\}$. The real numbers \mathbb{R} are also equipped with a Cauchy representation that is induced by a standard numbering α of all rational numbers. Besides the usual Cauchy representation there is also the so-called *naive Cauchy representation* δ_X^{n} that is defined exactly as δ_X but with $\text{dom}(\delta_X^{\text{n}}) := \{p \in \mathbb{N}^{\mathbb{N}} : (\alpha p(n))_{n \in \mathbb{N}} \text{ converges}\}$.

Given a computable metric space (X, d, α) we define the *open ball* $B_{(c,r)} := B(\alpha(c), \bar{r}) := \{x \in X : d(x, \alpha(c)) < \bar{r}\}$ for every $c, r \in \mathbb{N}$, where $\langle i, j, k \rangle := \frac{i-j}{k+1}$ denotes the rational number encoded by $i, j, k \in \mathbb{N}$. By $W_i := \{n \in \mathbb{N} : \text{Turing machine } i \text{ halts on input } n\}$ we denote the usual numbering of c.e. subsets of \mathbb{N} , and by $U_i := \bigcup_{n \in W_i} B_n$ we denote a numbering of all c.e. open subsets of X . We can consider the set $\mathcal{O}(X)$ of open subsets as a represented space with representation $\delta_{\mathcal{O}(X)}(p) = \bigcup_{i \in \mathbb{N}} B_{p(i)}$. The c.e. open subsets are exactly the computable points in $\mathcal{O}(X)$ with this representation.

We can now generalize the limit map and the Turing jump as defined in Example 2.2 to arbitrary computable metric spaces.

Definition 5.1 (Limit and Turing jump). Let X be a computable metric space. We define:

- (1) The *limit map* of X by $\lim_X : \subseteq X^{\mathbb{N}} \rightarrow X, (x_n)_{n \in \mathbb{N}} \mapsto \lim_{n \rightarrow \infty} x_n$.
- (2) The *Turing jump* $J_X : X \rightarrow \mathbb{N}^{\mathbb{N}}$ of X by

$$J_X(x)(i) := \begin{cases} 1 & \text{if } x \in U_i \\ 0 & \text{otherwise} \end{cases}$$

for all $x \in X$ and $i \in \mathbb{N}$.

It is easy to see that both functions defined here are always limit computable.

Proposition 5.2. \lim_X and J_X are limit computable for every computable metric space X .

Proof. Let δ_X be the Cauchy representation of the computable metric space (X, d, α) . Given a $p = \langle p_0, p_1, p_2, \dots \rangle \in \mathbb{N}^{\mathbb{N}}$ with a sequence $(p_n)_{n \in \mathbb{N}}$ of names of points $x_n := \delta_X(p_n)$ such that $x := \lim_{n \rightarrow \infty} x_n$ exists, we need to devise a limit computation that yields a $q \in \text{dom}(\delta_X)$ with $\delta_X(q) = x$. For every $i, k \in \mathbb{N}$ the property

$$(\exists n > i) d(\delta_X(p_n), \delta_X(p_i)) > 2^{-k-2}$$

is c.e. in p and k, i . Hence, with the help of $J(p)$ we can decide this property, and hence for each $k \in \mathbb{N}$ we can systematically search for a $i = i_k \in \mathbb{N}$ such that it fails. Such an i_k must exist since x_n converges to x . We can assume that the sequence $(i_k)_{k \in \mathbb{N}}$ is strictly monotone increasing. For each $k \in \mathbb{N}$ we let $q(k) := p_{i_k}(k+2)$. Then for $n > k$

$$\begin{aligned} d(\alpha q(n), \alpha q(k)) &\leq d(\alpha p_{i_n}(n+2), \delta_X(p_{i_n})) + d(\delta_X(p_{i_n}), \delta_X(p_{i_k})) + d(\delta_X(p_{i_k}), \alpha p_{i_k}(k+2)) \\ &\leq 2^{-n-2} + 2^{-k-2} + 2^{-k-2} < 2^{-k} \end{aligned}$$

and $\delta_X(q) = \lim_{k \rightarrow \infty} \alpha p_{i_k}(k+2) = \lim_{k \rightarrow \infty} x_k = x$. By Theorem 2.14 it follows that \lim_X is limit computable.

J_X can easily be computed on a limit Turing machine. Given a name of an input $x \in X$ one produces for each output position $J_X(x)(i)$ the default value 0 and simultaneously one tries to verify $x \in U_i$. As soon as $x \in U_i$ can be confirmed, the i -th output value has to be changed to 1, which happens at most once for each position i . \square

This allows us to transfer the limit normal form theorem (Theorem 2.6) to computable metric spaces. For $f, s : \subseteq X \rightrightarrows Y$ we write $s \sqsubseteq f$ if $s(x) \subseteq f(x)$ for all $x \in \text{dom}(f)$ and $\text{dom}(f) \subseteq \text{dom}(s)$. In this situation we call s a *solution* or *selector* of f .

Theorem 5.3 (Limit normal form). *Let X be a represented space, Y a computable metric space and $f : \subseteq X \rightrightarrows Y$ a problem. Then the following are equivalent:*

- (1) $f : \subseteq X \rightrightarrows Y$ is limit computable,
- (2) $\lim_Y \circ g \sqsubseteq f$ for some computable $g : \subseteq X \rightrightarrows Y^{\mathbb{N}}$.

Proof. That (2) implies (1) follows since \lim_Y is limit computable by Proposition 5.2. We need to prove that (1) implies (2). Let $f : \subseteq X \rightrightarrows Y$ be limit computable. We use the Cauchy representation δ_Y of Y . Then f has a limit computable realizer $F : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$. By the limit normal form theorem (Theorem 2.6) there exists a computable $G : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ such that $F = \lim \circ G$. Now we define a function $H : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ by $H(p) = \langle r_0, r_1, r_2, \dots \rangle$, provided that $G(p) = \langle q_0, q_1, q_2, \dots \rangle$, where each r_i is determined as follows. We choose $r_1 := r_0 := (q_0(0), q_0(0), \dots)$ and if $i > 1$ then for each $k \in \{2, \dots, i\}$ we check, which of the following two conditions is recognized first:

- (1) $(\forall j \in \{1, \dots, k-1\}) d(\alpha q_i(k), \alpha q_i(j)) < 2^{-j+1}$,
- (2) $(\exists j \in \{1, \dots, k-1\}) d(\alpha q_i(k), \alpha q_i(j)) > 2^{-j}$.

Depending on which condition is recognized first, we choose

- (1) $r_i := (q_i(1), q_i(2), \dots, q_i(i), q_i(i), \dots)$, if for all $k \leq i$ the first condition is recognized first,
- (2) $r_i := (q_i(1), q_i(2), \dots, q_i(k-1), q_i(k-1), \dots)$, if $k \leq i$ is minimal such that the second condition is recognized first.

We note that $r_i \in \text{dom}(\delta_Y)$ for all i and if q_i has a prefix of length $k \leq i$ that is a valid prefix of a name in $q \in \text{dom}(\delta_Y)$, then $d(\delta_Y(r_i), \delta_Y(q)) \leq 2^{-k+1}$. Altogether, H is computable, and the definition ensures that $\text{range}(H) \subseteq \text{dom}(\delta_Y^{\mathbb{N}})$ and $\lim_Y \circ \delta_Y^{\mathbb{N}} \circ H = \delta_Y \circ \lim \circ G = \delta_Y \circ F$. This proves that H is a realizer of a computable multi-valued function $g : \subseteq X \rightrightarrows Y^{\mathbb{N}}$ with $\lim_Y \circ g \sqsubseteq f$. \square

For completeness we formulate the result also for single-valued f . We note that the problem g still needs to be multi-valued in general.

Corollary 5.4 (Limit normal form). *Let X be a represented space, Y a computable metric space and $f : \subseteq X \rightarrow Y$ a map. Then the following are equivalent:*

- (1) $f : \subseteq X \rightarrow Y$ is limit computable,
- (2) $f = \lim_Y \circ g$ for some computable $g : \subseteq X \rightrightarrows Y^{\mathbb{N}}$.

As a corollary we obtain the following conclusion on pointwise limits of computable sequences of functions.

Corollary 5.5 (Pointwise limit). *Let X be a represented space and Y a computable metric space. Let $(g_n)_{n \in \mathbb{N}}$ be a computable sequence in $\mathcal{C}(X, Y)$, i.e., a computable sequence of computable functions $g_n : X \rightarrow Y$. Then $f : X \rightarrow Y$, defined by $f(x) := \lim_{n \rightarrow \infty} g_n(x)$ for all $x \in X$, is limit computable.*

We note that in general one cannot expect that all limit computable f can be obtained as the pointwise limit of a computable sequence of single-valued g (as in the case of Baire space, see Corollary 2.7). We obtain, however, a characterization of limit computable points in computable metric spaces.

Corollary 5.6 (Limit computable points). *Let X be a computable metric space. Then $x \in X$ is limit computable (in the sense that it has a limit computable name) if and only if there is a computable sequence $(x_n)_{n \in \mathbb{N}}$ such that $x = \lim_{n \rightarrow \infty} x_n$.*

We recall that a computable metric space (X, d) is called *computably compact* if $\{X\}$ is c.e. open in $\mathcal{O}(X)$. We note that for computably compact metric spaces (X, d) the function space $\mathcal{C}(X) = \mathcal{C}(X, \mathbb{R})$ with the metric induced by the *uniform norm* $\|f\| := \sup_{x \in X} |f(x)|$ is a computable metric space again. As a dense subset one can use, for instance, the set of rational polynomials in the *distance functions* $d_x : X \rightarrow \mathbb{R}, y \mapsto d(x, y)$. This space $\mathcal{C}(X)$ is then computably isomorphic to the space obtained by using the function space representation $\delta_{\mathcal{C}(X, \mathbb{R})}$, see [3]. If we apply Corollaries 4.13 and 5.6 to the space $\mathcal{C}(X)$, then we obtain the following result.

Corollary 5.7 (Uniform limit). *Let X be a computably compact computable metric space. Then $f : X \rightarrow \mathbb{R}$ is computable relative to the halting problem if and only if there is a computable sequence $(f_n)_{n \in \mathbb{N}}$ of computable functions $f_n : X \rightarrow \mathbb{R}$ that converges uniformly to f , i.e., such that $\lim_{n \rightarrow \infty} \|f - f_n\| = 0$.*

A characterization similar to Corollary 5.4 was obtained for effectively Σ_2^0 -measurable functions in [6, Theorem 9.5] (we refer the reader to [6] for the definition of measurability, as we are not going to use these notions here any further). This yields the following conclusion.

Corollary 5.8 (Borel measurability). *Let X, Y be computable metric spaces. Then a function $f : X \rightarrow Y$ is limit computable if and only if it is effectively Σ_2^0 -measurable.*

Corollary 5.4 can also be applied to the naive Cauchy representation since $\delta_X^n = \lim_X \circ \widehat{\alpha}$, where $\widehat{\alpha} : \mathbb{N}^{\mathbb{N}} \rightarrow X^{\mathbb{N}}, p \mapsto (\alpha p(0), \alpha p(1), \alpha p(2), \dots)$ is the parallelization of α that is computable with respect to the Cauchy representation δ_X . We obtain the following result that generalizes corresponding results for the spaces $X = \mathbb{R}$ and $X = \mathcal{C}([0, 1]^n)$ by Ziegler [56, Propositions 2.5 and 2.7].

Corollary 5.9 (Naive Cauchy representation as jump). $\delta_X^n \equiv \lim_X \circ \delta_X^{\mathbb{N}} \equiv \delta'_X$ for every computable metric space X .

Proof. With the help of Theorem 4.5 and Corollary 5.4 one obtains $\delta_X^n \leq \delta'_X$. The reduction $\delta'_X \leq \lim_X \circ \delta_X^{\mathbb{N}}$ follows from the same theorems applied to the identity $\text{id} : X' \rightarrow X$. The remaining reduction $\lim_X \circ \delta_X^{\mathbb{N}} \leq \lim_X \circ \widehat{\alpha}$ is easy to prove directly, by choosing a diagonal sequence. \square

Together with Theorem 4.8 we obtain a proof of the following result from [12, Theorem 22].

Corollary 5.10 (B. and Hertling 2002). *Let X, Y be computable metric spaces. The functions $f : \subseteq X \rightarrow Y$ that are continuous with respect to the naive Cauchy representations on X and Y are the usual continuous functions.*

Now we study the jump J_X . We first prove that J_X^{-1} is computable in general.

Proposition 5.11 (Inverse jump). $J_X^{-1} : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow X$ is computable for every computable metric space X .

Proof. We recall that a standard representation of X can be defined by $\delta(p) = x : \iff \text{range}(p) = \{n \in \mathbb{N} : x \in B_n\}$. It is easy to see that $\delta \leq \delta_X$, where δ_X denotes the Cauchy representation of X [52, Theorem 8.1.4]. Hence, it suffices to compute J_X^{-1} with respect to δ . Since $J_X(x)$ is the characteristic function of $\{n \in \mathbb{N} : x \in U_n\}$, it is straightforward to generate a list of all $n \in \mathbb{N}$ with $x \in B_n$, given $J_X(x)$. \square

Using this result we obtain the following form of the jump normal form theorem. Surprisingly we need a variant of Schröder's representation [43] of computable metric spaces that has compact fibers, which can be computed with respect to positive information. Such a representation was studied in [6].

Theorem 5.12 (Jump normal form). *Let X, Y be computable metric spaces and $f : \subseteq X \rightrightarrows Y$ a problem. Then the following are equivalent:*

- (1) $f : \subseteq X \rightrightarrows Y$ is limit computable,
- (2) $g \circ J_X \sqsubseteq f$ for some computable $g : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow Y$.

Proof. By Proposition 5.2 it is clear that $g \circ J_X$ is limit computable for every computable $g : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow Y$. Let now $f : \subseteq X \rightrightarrows Y$ be limit computable. By Theorem 5.3 there is a computable $h : \subseteq X \rightrightarrows Y^{\mathbb{N}}$ such that $\lim_Y \circ h \sqsubseteq f$. Let $\delta \equiv \delta_X$ be a representation of X that is equivalent to the Cauchy representation δ_X and that makes $\kappa_X : X \rightarrow \mathcal{K}_+(\mathbb{N}^{\mathbb{N}}), x \mapsto \delta^{-1}\{x\}$ computable, where $\mathcal{K}_+(\mathbb{N}^{\mathbb{N}})$ denotes the space of compact subsets of $\mathbb{N}^{\mathbb{N}}$ endowed with the representation $\delta_{\mathcal{K}_+(\mathbb{N}^{\mathbb{N}})}(p) = K : \iff \text{range}(p) = \{n \in \mathbb{N} : K \cap B_n \neq \emptyset\}$. Such a representation δ with compact fibers $\delta^{-1}\{x\}$ exists for every computable metric space [6, Lemma 6.3 (5)]. Let F be a computable realizer of h , i.e., $\lim_Y \delta_{Y^{\mathbb{N}}} F \sqsubseteq f \delta$. That κ_X is computable implies that the set

$$U := \{(x, n, k) \in X \times \mathbb{N}^2 : (\exists p \in \delta^{-1}\{x\})(\exists i > n) d_Y(\delta_Y(F(p)(i)), \delta_Y(F(p)(n))) > 2^{-k}\}$$

is c.e. open relative to $\text{dom}(f) \times \mathbb{N}^2$, and hence there is a computable function $r : \mathbb{N} \rightarrow \mathbb{N}$ such that $U_{r\langle n, k \rangle} \cap \text{dom}(f) = \{x \in X : (x, n, k) \in U\} \cap \text{dom}(f)$. Let $x \in \text{dom}(f)$ and $k \in \mathbb{N}$. Then for all $p \in \delta^{-1}\{x\}$ there is some $n \in \mathbb{N}$ such that for all $i > n$ we have $d_Y(\delta_Y(F(p)(i)), \delta_Y(F(p)(n))) \leq 2^{-k}$, because $(\delta_Y F(p)(n))_{n \in \mathbb{N}}$ is convergent. Since $\delta^{-1}\{x\}$ is compact, the number $n \in \mathbb{N}$ even exists uniformly for all $p \in \delta^{-1}\{x\}$. That is, for every $x \in \text{dom}(f)$ and $k \in \mathbb{N}$ there is some $n \in \mathbb{N}$ with $x \notin U_{r\langle n, k \rangle}$. With the help of $J_X(x)(r\langle n, k \rangle)$ we can actually check this condition, and hence we can compute an $s : \mathbb{N} \rightarrow \mathbb{N}$ such that for each $k \in \mathbb{N}$ we obtain $x \notin U_{r\langle s(k), k \rangle}$. By Proposition 5.11 we can also obtain a $p \in \mathbb{N}^{\mathbb{N}}$ with $\delta(p) = x$ from $J_X(x)$. Hence there is a computable $g : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow Y$ with $g \circ J_X(x) = \lim_{k \rightarrow \infty} \delta_Y(F(p)(s(k))) = \lim_Y \delta_Y F(p) \sqsubseteq f(x)$ for all $x \in \text{dom}(f)$, where the first limit is computable since it is effective (with speed of convergence 2^{-k} , see the proof of Proposition 5.2 for a detailed calculation). \square

Since J_X is injective, we can assume that $\text{dom}(g)$ is such that $\text{dom}(g \circ J_X) = \text{dom}(f)$. We formulate the characterization for single-valued functions as a corollary.

Corollary 5.13 (Jump normal form). *Let X, Y be computable metric spaces and $f : \subseteq X \rightarrow Y$ a function. Then the following are equivalent:*

- (1) $f : \subseteq X \rightarrow Y$ is limit computable,
- (2) $f = g \circ J_X$ for some computable $g : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow Y$.

Another consequence of Theorem 5.12 is that every multi-valued limit computable problem has a single-valued limit computable selector.

Corollary 5.14 (Selection). *Let X, Y be computable metric spaces. For every limit computable $f : \subseteq X \rightrightarrows Y$ there is a limit computable single-valued $g : \subseteq X \rightarrow Y$ with $g \sqsubseteq f$.*

Now we define 1-genericity for computable metric spaces.

Definition 5.15 (1-genericity). *Let X be a computable metric space. Then we call $x \in X$ 1-generic if it is a point of continuity of J_X .*

It is easy to see that 1-generic points can be characterized as those points that avoid the boundaries of all c.e. open sets. By ∂U we denote the *boundary* of U .

Lemma 5.16 (1-genericity). *Let X be a computable metric space. Then $x \in X$ is 1-generic if and only if $x \in X \setminus \bigcup_{i=0}^{\infty} \partial U_i$.*

This characterization is equivalent to the definition of 1-genericity as it is used in computability theory and as it has been used in computable metric spaces before (see [30, Corollary 2.3], [10, Lemma 9.2] or [27, Definition 2.1]). By the Baire category theorem every complete metric space is comeager and since the set $\bigcup_{i=0}^{\infty} \partial U_i$ is meager, it follows that the 1-generics form a comeager G_δ -set in any complete computable metric space. In particular, in a complete computable metric space, there are many 1-generic points. We obtain the following characterization of 1-genericity.

Theorem 5.17 (1-genericity). *Let X be a computable metric space. Then $x \in X$ is 1-generic if and only if for every computable metric space Y , every limit computable $f : \subseteq X \rightarrow Y$ with $x \in \text{dom}(f)$ is continuous at x .*

Proof. For the “if” direction it is sufficient to consider $Y = \mathbb{N}^{\mathbb{N}}$ and $f = J_X : X \rightarrow \mathbb{N}^{\mathbb{N}}$. For the “only if” direction we assume that $f : \subseteq X \rightarrow Y$ is limit computable and $x \in \text{dom}(f)$.

Then by Corollary 5.13 there is a computable $g : \subseteq \mathbb{N}^{\mathbb{N}} \rightarrow X$ such that $f = g \circ J_X$. Hence, it follows that f is continuous at x . \square

The “if” direction of this theorem can be seen as a computability theoretic version of a well-known theorem of Baire that states that every Baire class 1 function has a comeager G_δ -set of continuity points [29, Theorem 24.14]. We note that the “only if” direction of Theorem 5.17 for the special case of $X = [0, 1]$ was also proved by Kuyper and Terwijn [30, Theorem 4.2].

Since \lim_X is not continuous at any point for computable metric spaces X with at least two points, it follows that no 1-generic sequence in such a space is convergent.

Corollary 5.18 (1-generic sequences). *Let X be a computable metric space with at least two points. Then every sequence $(x_n)_{n \in \mathbb{N}}$ that is a 1-generic point in $X^{\mathbb{N}}$ is not convergent.*

The following result generalizes Proposition 2.21 to computable metric spaces.

Proposition 5.19 (Jump on 1-generics). *The Turing jump operator $J_X : X \rightarrow \mathbb{N}^{\mathbb{N}}$ restricted to the set of 1-generic points $x \in X$ is computable relative to the halting problem for all computable metric spaces X .*

Proof. We consider the metric space (X, d) and we define the relations of *formal inclusion* and *formal disjointness* by

- (1) $B_{\langle c_1, r_1 \rangle} \triangleleft B_{\langle c_2, r_2 \rangle} : \iff d(\alpha(c_1), \alpha(c_2)) + \bar{r}_1 < \bar{r}_2$,
- (2) $B_{\langle c_1, r_1 \rangle} \bowtie B_{\langle c_2, r_2 \rangle} : \iff d(\alpha(c_1), \alpha(c_2)) > \bar{r}_1 + \bar{r}_2$.

These are rather relations between the numbers $c_1, r_1 \in \mathbb{N}$ and $c_2, r_2 \in \mathbb{N}$ than between the corresponding balls, but they are denoted for convenience in the given way. It is clear that both relations are c.e. and that $B_j \triangleleft B_i$ implies $B_j \subseteq B_i$ and $B_j \bowtie B_i$ implies $B_j \cap B_i = \emptyset$. The sets

- (1) $A := \{(n, j) \in \mathbb{N}^2 : (\exists i \in W_n) B_j \triangleleft B_i\}$,
- (2) $B := \{(n, j) \in \mathbb{N}^2 : (\forall i \in W_n) B_j \bowtie B_i\}$

are both computable relative to $0'$. A 1-generic $x \in X$ satisfies $x \in X \setminus \bigcup_{n=0}^{\infty} \partial U_n$ and hence

- (1) $x \in U_n \iff (\exists j \in \mathbb{N})(x \in B_j \text{ and } (n, j) \in A)$,
- (2) $x \notin U_n \iff (\exists j \in \mathbb{N})(x \in B_j \text{ and } (n, j) \in B)$.

Thus, for each 1-generic x we can systematically search for some $j \in \mathbb{N}$ such that $x \in B_j$ and either $(n, j) \in A$ or $(n, j) \in B$ and the condition that holds tells us how to compute $J_X(x)(n)$. Hence, J_X restricted to 1-generics is computable relative to $0'$. \square

Another consequence of the jump normal form (Theorem 5.12) together with Proposition 5.19 is the following result that generalizes Corollary 2.22.

Corollary 5.20 (Limit computability on 1-generics). *Restricted to 1-generics every limit computable $F : \subseteq X \rightarrow Y$ on computable metric spaces X, Y is computable relative to the halting problem.*

Now the question appears how the property of being 1-generic relates to the property of having a 1-generic name. What we can prove in general is that atypical points have atypical names. As a preparation we need to show that admissible representations are hereditarily quotient maps. We recall that a representation δ of a topological space X is called *admissible* (with respect to this space) if and only if it is maximal with respect to the topological version \leq_t of the reducibility \leq among all continuous representations [44]. The

Cauchy representation δ_X is an example of an admissible representation [52, Theorem 8.1.4], and hence any representation $\delta \equiv \delta_X$ is also admissible. It follows from the second half of the proof of [44, Theorem 4] that admissible representations are hereditary quotient maps in the following sense.

Lemma 5.21 (Hereditary quotients). *Let δ be an admissible representation of a topological space X , and let Y be another topological space. Let $f : \subseteq X \rightarrow Y$ be a function and $x \in \text{dom}(f)$. If $f\delta$ is continuous at all $p \in \delta^{-1}\{x\}$, then f is sequentially continuous at x .*

Since every metric space X is sequential, we can replace sequential continuity by ordinary continuity in this case. Then the property expresses what is usually called a *hereditary quotient map* or a *pseudo-open map* [2].

Corollary 5.22 (Hereditary quotients). *Every admissible representation of a sequential topological space is a hereditary quotient map (or equivalently, is pseudo-open).*

Using Lemma 5.21 we can now prove the following result, which shows that atypical points have atypical names with respect to every representation in the equivalence class of the Cauchy representation.

Proposition 5.23 (Non-generic names). *Let X be a computable metric space with Cauchy representation δ_X and $x \in X$. Let $\delta \equiv \delta_X$ be another representation of X . If x is not 1-generic, then there is a $p \in \mathbb{N}^{\mathbb{N}}$ with $\delta(p) = x$ that is not 1-generic.*

Proof. By Propositions 5.2 and 2.14 there is some computable F such that $J_X\delta = FJ$. Let x be such that all names $p \in \delta^{-1}\{x\}$ are 1-generic. Then FJ is continuous at all these p and hence so is $J_X\delta$. By Lemma 5.21 this implies that J_X is continuous at x , hence x is 1-generic. \square

This result captures as much as can be said about the relation of arbitrary points and their names with respect to genericity in arbitrary computable metric spaces. It is easy to see that in the equivalence class of the Cauchy representation δ_X there are always representations without 1-generic names: one can for instance define $\delta \equiv \delta_X$ such that names are always zero in every second component and hence not 1-generic. But even if we restrict ourselves to total $\delta \equiv \delta_X$ for complete metric spaces X , then there are spaces such as $X = \mathbb{N}$ in which all points are 1-generic. Hence having a non 1-generic name does not imply being not 1-generic in general. If we consider the case of the reals $X = \mathbb{R}$ and reasonable total variants of the Cauchy representation (such as the one from [10, Lemma 6.1]), then there are syntactic properties of names such as: every component of the name encodes a rational number with even denominator. This property constitutes a co-c.e. closed set $A \subseteq \mathbb{N}^{\mathbb{N}}$ that is nowhere dense, i.e., $\partial A = A$, which means that all points in A are not 1-generic. On the other hand, every real x , in particular every 1-generic real, has a name $p \in A$. Thus having a non 1-generic name does not imply being not 1-generic for such a representation. For special types of spaces and special representations one can provide further results along these lines. For instance Kuyper and Terwijn proved that an irrational number $x \in [0, 1]$ is 1-generic if and only if its unique binary expansion is 1-generic [30, Proposition 2.5].

6. LIMIT COMPUTABILITY AND COMPUTABILITY RELATIVE TO THE HALTING PROBLEM

It is clear that every function that is computable relative to the halting problem is also limit computable. This observation even holds uniformly in the sense specified in the

following corollary. The corollary is a consequence of Theorem 4.12 (3) since $\text{id} : X \rightarrow X'$ is computable.

Corollary 6.1 (Limit computability of functions computable relative to the halting problem). *$\text{id} : \mathcal{C}(X, Y)' \rightarrow \mathcal{C}(X, Y')$ is computable for all represented spaces X, Y .*

For some spaces, such as $X = \mathbb{N}$, this identity is even a computable isomorphism (see Corollary 4.14). However, it is clear that this does not happen for many other spaces X, Y since often $\mathcal{C}(X, Y)' \subsetneq \mathcal{C}(X, Y')$. The former set contains only continuous functions, the latter can contain discontinuous ones. Even restricted to continuous functions the identity in Corollary 6.1 is not a computable isomorphism in general.

We can, however, characterize the functions that are computable relative to the halting problems in terms of limit computable functions. For this purpose we need the notion of a modulus of continuity. Let (X, d_X) and (Y, d_Y) be metric spaces and $f : X \rightarrow Y$ a function. Then $m : \mathbb{N} \rightarrow \mathbb{N}$ is called a *modulus of continuity* of f at $x \in X$ if

$$d_X(x, y) < 2^{-m(n)} \implies d_Y(f(x), f(y)) < 2^{-n}$$

holds for all $y \in X$ and $n \in \mathbb{N}$. The following result is well-known (see, for instance, [3, Lemma 4.4.25]).

Proposition 6.2 (Modulus of continuity). *Let X, Y be computable metric spaces. Then*

$\text{Mod} : \mathcal{C}(X, Y) \times X \rightrightarrows \mathbb{N}^{\mathbb{N}}, (f, x) \mapsto \{m \in \mathbb{N}^{\mathbb{N}} : m \text{ is a modulus of continuity of } f \text{ at } x\}$
is computable.

We call $M : X \rightrightarrows \mathbb{N}^{\mathbb{N}}, x \mapsto \{m \in \mathbb{N}^{\mathbb{N}} : m \text{ is a modulus of continuity of } f \text{ at } x\}$ the *global modulus of continuity* of f . Using the previous proposition we can prove the following characterization.

Theorem 6.3 (Computability relative to the halting problem). *Let X, Y be computable metric spaces and let $f : X \rightarrow Y$ be a function. Then the following are equivalent:*

- (1) *f is computable relative to the halting problem,*
- (2) *f is limit computable and continuous and its global modulus of continuity is computable relative to the halting problem.*

Proof. Let $f : X \rightarrow Y$ be computable relative to the halting problem. Then $f : X' \rightarrow Y'$ is computable by Theorem 4.8, and hence f is computable as a point in $\mathcal{C}(X, Y)'$ by Theorem 4.12. In particular, f is also limit computable (see Corollary 6.1). By Theorem 4.12 and Proposition 4.4 the problem Mod from Proposition 6.2 is computable with type $\text{Mod} : \mathcal{C}(X, Y)' \times X' \rightrightarrows (\mathbb{N}^{\mathbb{N}})'$, and hence the global modulus of f is computable with type $M : X' \rightrightarrows (\mathbb{N}^{\mathbb{N}})', x \mapsto \text{Mod}(f, x)$. By Theorem 4.8 this means that M is computable relative to the halting problem.

Let now f be limit computable and continuous with a global modulus of continuity M that is computable relative to the halting problem. We consider the computable metric spaces (X, d_X, α) and (Y, d_Y) . If f is limit computable, then also $f \circ \alpha : \mathbb{N} \rightarrow Y$ is limit computable and hence computable relative to the halting problem by Corollary 4.14. Given $x \in X$ and $k \in \mathbb{N}$, we can compute with the help of the halting problem a modulus of continuity $m : \mathbb{N} \rightarrow \mathbb{N}$ of f at x , and we can compute an $n \in \mathbb{N}$ with $d_X(\alpha(n), x) < 2^{-m(k)}$, which implies $d_Y(f\alpha(n), f(x)) < 2^{-k}$. Since $f\alpha(n)$ can be computed with the help of the halting problem, we have found an approximation of $f(x)$ with precision 2^{-k} with the help of the halting problem. This means that f is computable relative to the halting problem. \square

This result tells us exactly which additional condition beyond continuity a limit computable function $f : X \rightarrow Y$ must satisfy in order to be computable relative to the halting problem. If a function is limit computable and just continuous on metric spaces, then a hyperarithmetical oracle is sufficient to compute it. In fact, we can prove an even stronger result.⁶

Theorem 6.4 (Effective Borel measurability and continuity). *Let X, Y be computable metric spaces and let X be complete. Let $A \subseteq X$ be an effective Σ_1^1 -subset and let $f : A \rightarrow Y$ be a function. If f is effectively Borel measurable and continuous, then f is computable relative to some hyperarithmetical oracle (i.e., an oracle in Δ_1^1).*

Proof. We use a standard numbering $(U_n^p)_{n \in \mathbb{N}}$ of the sets $U_n^p \subseteq Y$ that are c.e. relative to $p \in \mathbb{N}^{\mathbb{N}}$. We omit the upper index in order to denote the unrelativized c.e. open subsets. For any c.e. open set $U_n \subseteq Y$ we have that $f^{-1}(U_n)$ is effectively Δ_1^1 and open in A , i.e., we can effectively find some Δ_1^1 -set D_n in X and there is an open set $V \subseteq X$ such that $f^{-1}(U_n) = V \cap A = D_n \cap A$. In particular, $f^{-1}(U_n)$ and $A \setminus f^{-1}(U_n)$ are effectively Σ_1^1 in X and separated by the open set V . Then by the Louveau separation theorem [32, Theorem B, page 369] the sets $f^{-1}(U_n)$ and $A \setminus f^{-1}(U_n)$ are also separated by a $\Sigma_1^{0,p}$ -set for some hyperarithmetical $p \in \mathbb{N}^{\mathbb{N}}$ (i.e., by a c.e. open set relative to p). Hence $f^{-1}(U_n) = U_k^p \cap A$ for some $k \in \mathbb{N}$ and some hyperarithmetical $p \in \mathbb{N}^{\mathbb{N}}$. Now we consider the function $g : \mathbb{N} \rightarrow \mathbb{N}^{\mathbb{N}}, n \mapsto \langle k, p \rangle$. The property $B := \{\langle n, k, p \rangle \in \mathbb{N}^{\mathbb{N}} : f^{-1}(U_n) = U_k^p \cap A\}$ is an effective Π_1^1 -property, since

$$\langle n, k, p \rangle \in B \iff (\forall x \in X)((x \notin U_k^p \cap A \vee x \in D_n) \wedge (x \notin f^{-1}(U_n) \vee x \in U_k^p)).$$

Hence, by the effective Π_1^1 -uniformization theorem [33, Theorem 4E.4] the function g is a total Π_1^1 -function, hence in Δ_1^1 . Clearly, f is computable relative to g . \square

As an immediate corollary we obtain the following result.

Corollary 6.5 (Limit computability and continuity). *Let X, Y be computable metric spaces and let X be complete. If $f : X \rightarrow Y$ is limit computable and continuous, then f is computable relative to some hyperarithmetical oracle (i.e., an oracle in Δ_1^1).*

For uniformly continuous functions we can improve this result. In this case even the jump of the halting problem suffices to compute the function.

Theorem 6.6 (Limit computability and uniform continuity). *Let X, Y be computable metric spaces. If $f : X \rightarrow Y$ is limit computable and uniformly continuous, then f is computable relative to $0''$.*

Proof. We consider the computable metric spaces (X, d_X, α) and (Y, d_Y) . As in the proof of Theorem 6.3 we obtain that $f \circ \alpha : \mathbb{N} \rightarrow Y$ is computable relative to $0'$. Additionally, the set

$$A := \{(n, k) \in \mathbb{N}^2 : (\exists i, j \in \mathbb{N})(d_X(\alpha(i), \alpha(j)) < 2^{-n} \text{ and } d_Y(f\alpha(i), f\alpha(j)) > 2^{-k})\}$$

is c.e. relative to $0'$ and hence computable relative to $0''$. Since f is uniformly continuous, it follows that for each $k \in \mathbb{N}$ there are only finitely many $n \in \mathbb{N}$ with $(n, k) \in A$. Hence, for each $k \in \mathbb{N}$ we can compute some $m(k)$ with $(m(k), k+1) \notin A$ with the help of $0''$ and $m : \mathbb{N} \rightarrow \mathbb{N}$ is a uniform modulus of continuity of f (i.e., at every $x \in X$). We can proceed as in the proof of Theorem 6.3 to show that f is computable relative to $0''$. \square

⁶This stronger result and its proof have been kindly suggested by an anonymous referee.

It is clear that this theorem also holds for compact computable metric spaces X and continuous and limit computable f , since any such f is automatically uniformly continuous.

Corollary 6.7 (Limit computability and uniform continuity). *Let X, Y be computable metric spaces and let X be compact. If $f : X \rightarrow Y$ is limit computable and continuous, then f is computable relative to $0''$.*

We could also prove such a result for effectively locally compact X and continuous and limit computable f , since it is sufficient to obtain the modulus of continuity for a suitable neighborhood of the input.

We will show that Theorem 6.6 gives the best possible condition and $0''$ cannot be improved to $0'$. In the following we will see some examples of continuous functions that are limit computable but not computable relative to the halting problem. For these examples we use the set $\text{Fin} := \{n \in \mathbb{N} : W_n \text{ finite}\}$, which is known to be Σ_2^0 -complete in the arithmetical hierarchy [47, Theorem 4.3.2], and hence it is not computable relative to the halting problem. Its complement $\text{Inf} := \mathbb{N} \setminus \text{Fin}$ is Π_2^0 -complete and hence not even c.e. relative to the halting problem. By $|A|$ we denote the *cardinality* of a set A .

Proposition 6.8. *There exists a uniformly continuous function $f : 2^{\mathbb{N}} \rightarrow \mathbb{R}$ that is computable with finitely many mind changes and hence low and limit computable, but that is not computable relative to the halting problem.*

Proof. We define $f : 2^{\mathbb{N}} \rightarrow \mathbb{R}$ by

$$f(p) := \begin{cases} 2^{-n} & \text{if } 0^n 1^{|W_n|+1} 0 \sqsubseteq p \text{ for some } n \in \text{Fin} \\ 0 & \text{otherwise} \end{cases}.$$

Then f is continuous and hence uniformly continuous, since $2^{\mathbb{N}}$ is compact. We use a fixed computable enumeration $e_n : \mathbb{N} \rightarrow \mathbb{N}$ of W_n , i.e., $\text{range}(e_n) = W_n$. The following algorithm shows that f is computable with finitely many mind changes:

- (1) We start producing the default output 0 as long as no other value is determined in the next step.
- (2) As soon as a prefix of the input p of form $0^n 1^{k+1} 0 w$ with $w \in \{0, 1\}^m$ is known, we check if there are exactly k different values among the numbers $e_n(0), \dots, e_n(m)$. If so, then the output is replaced by 2^{-n} , otherwise by 0.

This algorithm describes a computation with finitely many mind changes, since for any fixed input p , the output value changes at most two times. In particular, f is low and limit computable. Let us assume that f is computable relative to the halting problem. Then by Corollary 5.7 there exists a computable sequence $(f_k)_{k \in \mathbb{N}}$ of computable functions $f_k : 2^{\mathbb{N}} \rightarrow \mathbb{R}$ such that $\lim_{k \rightarrow \infty} \|f_k - f\| = 0$. Since the sequence $(K_n)_{n \in \mathbb{N}}$ with $K_n := 0^n 1 2^{\mathbb{N}}$ is a computable sequence of computably compact sets, it follows that $(f_k(K_n))_{(k,n) \in \mathbb{N}}$ is a computable sequence of computably compact sets too [37]. Since the maximum is computable on computably compact sets by [52, Lemma 5.2.6], it follows that $(\max f_k(K_n))_{k \in \mathbb{N}}$ is a computable sequence, and it converges for each $n \in \mathbb{N}$ to $\max f(K_n)$. Hence, by

$$s_n := \lim_{k \rightarrow \infty} \max f_k(K_n) = \max f(0^n 1 2^{\mathbb{N}}) = \begin{cases} 2^{-n} & \text{if } n \in \text{Fin} \\ 0 & \text{otherwise} \end{cases}$$

we define a sequence of numbers $(s_n)_{n \in \mathbb{N}}$ that is limit computable and hence computable relative to the halting problem by Corollary 4.14. But this would imply that Fin is computable relative to the halting problem. Contradiction! \square

We can transfer this construction from Cantor space to the unit interval $[0, 1]$.

Proposition 6.9. *There exists a uniformly continuous function $f : [0, 1] \rightarrow \mathbb{R}$ that is computable with finitely many mind changes and hence low and limit computable, but that is not computable relative to the halting problem.*

Proof. We use the computable “triangle” function

$$\Delta : \mathbb{R} \rightarrow \mathbb{R}, x \mapsto \begin{cases} 1 - |x| & \text{if } -1 \leq x \leq 1 \\ 0 & \text{otherwise} \end{cases}$$

This function has the extreme values $f(-1) = f(1) = 0$ and $f(0) = 1$ and interpolates linearly between them otherwise. Outside of the interval $(-1, 1)$ the function f is identically zero. We define a computable double sequence $(\Delta_{n,k})_{n,k \in \mathbb{N}}$ of triangle functions $\Delta_{n,k} : [0, 1] \rightarrow \mathbb{R}$ scaled to the interval $I_{n,k} := (2^{-n-1}, 2^{-n-1} + 2^{-n-k-1})$ by

$$\Delta_{n,k}(x) := \Delta(2^{n+k+2}(x - 2^{-n-1}) - 1)$$

The function $\Delta_{n,k}$ is zero outside of $K_n := [2^{-n-1}, 2^{-n}]$, in fact even outside of the interval $I_{n,k}$. With growing k the triangle of $\Delta_{n,k}$ is compressed further on the left hand side of K_n . We define a continuous function $f : [0, 1] \rightarrow \mathbb{R}$ by

$$f(x) := \sum_{n \in \text{Fin}} 2^{-n} \Delta_{n, |W_n|}(x).$$

Similarly as in the proof of Proposition 6.8 one can show that f is computable with finitely many mind changes and not computable relative to the halting problem. For the latter part one considers the values $s_n := \max f(K_n)$. For the former part we proceed as follows with a given input $x \in [0, 1]$:

- (1) As long as we cannot exclude that $x = 2^{-n}$ for some $n \in \mathbb{N}$ we produce the value 0 as output.
- (2) As soon as we detect that $x \in (2^{-n-1}, 2^{-n})$ we consider the enumeration $e_n(0), e_n(1), \dots$ of W_n .
- (3) Whenever we find a new number k of distinct values in this enumeration, then we produce the value $2^{-n} \Delta_{n,k}(x)$ as output.

At any point x at most a finite number of mind changes is required since either the value k stabilizes or $\Delta_{n,k}(x) = 0$ for all sufficiently large k . □

We note that Proposition 6.9 cannot be strengthened such that the function f is additionally continuously differentiable (see Corollary 6.16). However, this can be achieved by using smooth versions of the triangle function Δ that are distributed all over the entire real line.

Proposition 6.10. *There exists a function $f : \mathbb{R} \rightarrow \mathbb{R}$ that is infinitely often differentiable, computable with finitely many mind changes and hence low and limit computable, but that is not computable relative to the halting problem.*

Proof. We use the computable “bump” function $\Delta : \mathbb{R} \rightarrow \mathbb{R}$

$$\Delta(x) := \begin{cases} e^{\frac{1}{1-x^2}} & \text{if } |x| < 1 \\ 0 & \text{otherwise} \end{cases},$$

which is infinitely often continuously differentiable, computable and zero outside of $(-1, 1)$. We define a computable double sequence $(\Delta_{n,k})_{n,k \in \mathbb{N}}$ of bump functions $\Delta_{n,k} : \mathbb{R} \rightarrow \mathbb{R}$ scaled to the interval $I_{n,k} := (n, n + 2^{-k-1})$ by

$$\Delta_{n,k}(x) := \Delta(2^{k+2}(x - n) - 1)$$

The function $\Delta_{n,k}$ is zero outside of $K_n := [n, n + 1]$, in fact even outside of the interval $I_{n,k}$. With growing k the triangle of $\Delta_{n,k}$ is compressed further on the left hand side of K_n . We define $f : \mathbb{R} \rightarrow \mathbb{R}$ by

$$f(x) := \sum_{n \in \text{Fin}} \Delta_{n, |W_n|}(x).$$

Similarly as in the proof of Proposition 6.9 one can show that f is computable with finitely many mind changes and not computable relative to the halting problem. Additionally, f is infinitely often differentiable, since the $\Delta_{n,k}$ are so. \square

We can also produce a similar counterexample of type $f : 2^{\mathbb{N}} \rightarrow \mathbb{S}$, where \mathbb{S} denotes *Sierpiński space* $\mathbb{S} = \{0, 1\}$ that is represented by $\delta_{\mathbb{S}}(p) = 0 : \iff p = \widehat{0}$.

Proposition 6.11. *There exists a continuous function $f : 2^{\mathbb{N}} \rightarrow \mathbb{S}$ that is computable with finitely many mind changes and hence low and limit computable, but that is not computable relative to the halting problem.*

Proof. We define $f : 2^{\mathbb{N}} \rightarrow \mathbb{S}$ by

$$f(p) := \begin{cases} 0 & \text{if } p = \widehat{0} \\ 0 & \text{if } 0^n 1^{|W_n|+1} 0 \sqsubseteq p \text{ for some } n \in \text{Fin} \\ 1 & \text{otherwise} \end{cases} .$$

The sets $A_n := 0^n 1^{|W_n|+1} 0 2^{\mathbb{N}}$ are clopen for every $n \in \text{Fin}$ and the set $A := \{\widehat{0}\} \cup \bigcup_{n \in \text{Fin}} A_n$ is closed. Hence f is the characteristic function χ_U of the open set $U = 2^{\mathbb{N}} \setminus A$ and hence continuous. The following algorithm shows that f is computable with finitely many mind changes:

- (1) We start producing the default output 0 as long as no other value is determined in the next step.
- (2) As soon as a prefix of the input p of form $0^n 1^{k+1} 0^{m+1}$ is known, we check if there are exactly k different values among the numbers $e_n(0), \dots, e_n(m)$. If so, then the output is replaced by 0, otherwise by 1.

This algorithm describes a computation with finitely many mind changes, since for any fixed input p , the output value changes at most three times. In particular, f is limit computable.

Let us assume that $f = \chi_U$ is also computable relative to the halting problem. We note that the map $\mathcal{O}(2^{\mathbb{N}}) \rightarrow \mathcal{C}(2^{\mathbb{N}}, \mathbb{S}), V \mapsto \chi_V$ is a computable isomorphism, and hence U is a computable point in $\mathcal{O}(2^{\mathbb{N}})'$. The map $\forall_K : \mathcal{O}(2^{\mathbb{N}}) \rightarrow \mathbb{S}$ with $\forall_K(V) = 1 \iff K \subseteq V$ is computable for every computably compact set $K \subseteq 2^{\mathbb{N}}$, and the map $K \mapsto \forall_K$ is a computable isomorphism between the space of compact subsets of $2^{\mathbb{N}}$ and $\mathcal{C}(\mathcal{O}(2^{\mathbb{N}}), \mathbb{S})$ [37]. Since $(K_n)_{n \in \mathbb{N}}$ with $K_n := 0^n 1 2^{\mathbb{N}}$ is a computable sequence of computably compact sets, we obtain that \forall_{K_n} is computable uniformly in n . Hence $n \mapsto \forall_{K_n}(U)$ is computable relative to the halting problem by Corollary 4.14. This is a contradiction since

$$\forall_{K_n}(U) = 1 \iff 0^n 1 2^{\mathbb{N}} \subseteq U \iff n \in \text{Inf}$$

and Inf is not c.e. relative to the halting problem. \square

Since a set $U \subseteq 2^{\mathbb{N}}$ is c.e. open (relative to some oracle q) if and only if $\chi_U : 2^{\mathbb{N}} \rightarrow \mathbb{S}$ is computable (relative to q) and $F \subseteq 2^{\mathbb{N}}$ is a *computable* Σ_2^0 -set in the Borel hierarchy (or, equivalently, a *computable* F_σ -set) if and only if $\chi_F : 2^{\mathbb{N}} \rightarrow \mathbb{S}'$ is computable [38, Proposition 26], the previous proposition can also be rephrased in the following way.

Corollary 6.12. *There is a computable Σ_2^0 -set $U \subseteq 2^{\mathbb{N}}$ in the Borel hierarchy that is open, but not c.e. open relative to the halting problem.*

A similar result for \mathbb{R}^n was proved by Ziegler [54, Theorem 4.4. a)]. There is even a much stronger counter example for $2^{\mathbb{N}}$. There is a Π_2^0 -set $A \subseteq 2^{\mathbb{N}}$ that is a singleton (its only member being the ω -jump of the empty set) such that this set is closed but not a Π_1^0 -set relative to $0^{(n)}$ for any $n \in \mathbb{N}$ (see [35, Proposition 1.8.62 and Exercise 1.8.67], [42, § 15.1 Theorem XII] and [31]).

Proposition 6.13 (Kuznécov and Trahténbrot 1955). *There is a Σ_2^0 -set $A \subseteq 2^{\mathbb{N}}$ with a singleton complement that is not c.e. open relative to $0^{(n)}$ for any $n \in \mathbb{N}$.*

Further results along these lines can be found in [14]. If we re-translate this back to functions, then we obtain the following corollary.

Corollary 6.14. *There is a continuous and limit computable $f : 2^{\mathbb{N}} \rightarrow \mathbb{S}$ that is not computable relative to $0^{(n)}$ for any $n \in \mathbb{N}$.*

This also shows that Corollary 6.7 cannot be generalized to arbitrary spaces Y and that the function f in Corollary 6.14 cannot have a realizer $F : 2^{\mathbb{N}} \rightarrow \mathbb{N}^{\mathbb{N}}$ that is simultaneously continuous and limit computable, even though it has separate realizers with each of these properties individually.

There are some classes of functions for which the condition on the modulus of continuity in Theorem 6.3 is automatically satisfied. The first example is the class of Hölder continuous functions. Let (X, d_X) and (Y, d_Y) be metric spaces. Then $f : X \rightarrow Y$ is called *Hölder continuous* with constant $L > 0$ and exponent $\alpha \in (0, 1]$ if

$$d_Y(f(x), f(y)) \leq L d_X(x, y)^\alpha$$

holds for all $x, y \in X$. A function is called *Lipschitz continuous* with constant $L > 0$ if it is Hölder continuous with constant L and exponent $\alpha = 1$. Since every Hölder continuous function automatically has a computable constant L (perhaps slightly larger than necessary) and a computable exponent α (perhaps slightly smaller than necessary and correct at least for small distances $d_X(x, y) < 1$), it follows that it automatically has a computable global modulus of continuity. Hence we obtain the following corollary of Theorem 6.3.

Corollary 6.15 (Hölder continuity). *Let X and Y be computable metric spaces, and let $f : X \rightarrow Y$ be Hölder continuous. Then f is limit computable if and only if f is computable relative to the halting problem.*

Restricted to the set of Hölder continuous functions with a fixed Lipschitz constant L and a fixed exponent α the map in Corollary 6.1 is even a computable isomorphism, as the proof of “(2) \implies (1)” of Theorem 6.3 is uniform. We note that even restricted to Lipschitz continuous functions the notions of a low function and a function that is computable relative to a low oracle are incomparable by Proposition 4.15.

In the table in Figure 2 we summarize which oracle classes are sufficient to compute a limit computable function $f : X \rightarrow Y$ on complete computable metric spaces that satisfies extra continuity assumptions.

f limit computable and	oracle class
continuous	Δ_1^1
uniformly continuous	Σ_2^0
Lipschitz or Hölder continuous	Σ_1^0

Figure 2: Oracle classes sufficient to compute $f : X \rightarrow Y$.

Since every continuously differentiable function $f : [0, 1] \rightarrow \mathbb{R}$ is Lipschitz continuous with Lipschitz constant $\max f'[0, 1]$, we obtain the following conclusion (here it is important that $[0, 1]$ is compact, and the conclusion does not hold for functions of type $f : \mathbb{R} \rightarrow \mathbb{R}$ by Proposition 6.10).

Corollary 6.16 (Continuously differentiable functions). *Let $f : [0, 1] \rightarrow \mathbb{R}$ be continuously differentiable. Then f is limit computable if and only if f is computable relative to the halting problem.*

We can also draw some conclusions on linear operators. We recall that a computable normed space is just a separable normed space such that the induced metric space is computable and makes the algebraic operations computable. By the Borel graph theorem of Schwartz [45] (see also [1, Corollary II.10.4]) every Borel measurable linear operator $T : X \rightarrow Y$ from a Banach space X to a normed space Y is continuous, i.e., bounded. Using Theorem 6.4 we obtain the following conclusion on effectively Borel measurable operators.

Corollary 6.17 (Borel measurable linear functions). *Let X be a computable Banach space, let Y be computable normed spaces, and let $T : X \rightarrow Y$ be linear. Then T is effectively Borel measurable if and only if it is computable relative to a hyperarithmetical oracle (i.e., a Δ_1^1 oracle).*

Every linear bounded operator is automatically Lipschitz continuous. In the special case of limit computability Corollary 6.15 yields the following.

Corollary 6.18 (Limit computable linear functions). *Let X be a computable Banach space, Y be computable normed spaces, and let $T : X \rightarrow Y$ be linear. Then T is limit computable if and only if T is computable relative to the halting problem.*

7. APPLICATIONS

In this section we discuss a number of simple applications of the tools that we have developed in the previous sections. Some of the results that we derive are known, but our techniques provide very simple proofs. Other results are new.

We start with considering distance functions. For every subset $A \subseteq X$ of a metric space (X, d) we define the *distance function* $d_A : X \rightarrow \mathbb{R}, x \mapsto \inf_{y \in A} d(x, y)$. Every distance function d_A satisfies $|d_A(x) - d_A(y)| \leq d(x, y)$ for all $x, y \in X$, which means that every distance function is Lipschitz continuous with Lipschitz constant 1.

Corollary 7.1 (Distance functions). *Let X be a computable metric space and $A \subseteq X$. The distance function $d_A : X \rightarrow \mathbb{R}$ is limit computable if and only if it is computable relative to the halting problem.*

We recall that a problem $f : X \rightarrow \mathbb{R}$ is called *lower semi-computable* if it is computable as a function $f : X \rightarrow \mathbb{R}_{<}$ with the real numbers $\mathbb{R}_{<}$ equipped with the lower Dedekind cut representation, and likewise f is called *upper semi-computable* if it is computable as a function $f : X \rightarrow \mathbb{R}_{>}$, where the real numbers $\mathbb{R}_{>}$ are equipped with the upper Dedekind cut representation. Since it is clear that $\text{id} : \mathbb{R}_{<} \rightarrow \mathbb{R}$ and $\text{id} : \mathbb{R}_{>} \rightarrow \mathbb{R}$ are limit computable [8, Proposition 3.7], we obtain the following conclusion.

Corollary 7.2 (Semi-computable functions). *Every lower or upper semi-computable function $f : X \rightarrow \mathbb{R}$ is limit computable.*

By $f^{\circ n}$ we denote the n -fold composition of f with itself, i.e., $f^{\circ 0} = \text{id}$, $f^{\circ 1} = f$, $f^{\circ 2} = f \circ f$, etc. It is easy to see that the *Mandelbrot set* $M := \{c \in \mathbb{C} : (\forall n \in \mathbb{N}) |f_c^{\circ n}(0)| \leq 2\}$, which is defined using the iteration function $f_c : \mathbb{C} \rightarrow \mathbb{C}, z \mapsto z^2 + c$, is co-c.e. closed (see [52, Exercise 5.1.32 (c)] and [24]), and hence its distance function $d_M : \mathbb{C} \rightarrow \mathbb{R}$ is lower semi-computable [13, Corollary 3.14 (2)]. As a corollary of Theorem 6.3 and Corollaries 7.1 and 7.2 we obtain the following.

Corollary 7.3 (Mandelbrot set). *The distance function $d_M : \mathbb{C} \rightarrow \mathbb{R}$ of the Mandelbrot set $M \subseteq \mathbb{C}$ is computable relative to the halting problem.*

Another application concerns the field of limit computable real numbers. By $(\mathbb{R}^{(n)})_c$ we denote the set of points that are computable in $\mathbb{R}^{(n)}$, i.e., computable with respect to the n -th jump of the Cauchy representation. We recall that a subfield of the reals is called *real algebraically closed* if the real-valued zeros of all non-constant polynomials with coefficients from the subfield are again in the subfield.

Proposition 7.4 (Field of real numbers). *The real numbers in $(\mathbb{R}^{(n)})_c$ form a real algebraically closed subfield of \mathbb{R} for every $n \in \mathbb{N}$.*

Proof. We note the following facts for $n \in \mathbb{N}$ and $a, b \in \mathbb{Q}$ with $a < b$ (see [52]):

- (1) The algebraic operations $+, -, \cdot : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$ are computable.
- (2) Division $\div : \subseteq \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$ is computable.
- (3) The constants $0, 1$ are computable real numbers.
- (4) The map $p : \mathbb{R}^{n+1} \rightarrow \mathcal{C}[a, b], (a_0, \dots, a_n) \mapsto (x \mapsto \sum_{i=0}^n a_i x^i)$ is computable.
- (5) There is a computable $Z_{[a,b]} : \subseteq \mathcal{C}[a, b] \rightarrow \mathbb{R}$ such that $f(Z_{[a,b]}(f)) = 0$ for every continuous $f : [a, b] \rightarrow \mathbb{R}$ that has exactly one zero (see [52, Corollary 6.3.5], where this was proved for $[a, b] = [0, 1]$; it is straightforward to generalize the proof).

It follows from (1)–(3) that \mathbb{R}_c is a computable subfield of \mathbb{R} and from (4) and (5) that this field is real algebraically closed as the zeros of all non-constant polynomials are isolated and can be computed if the coefficients are all computable. If we apply Proposition 4.4, then we obtain that all the operations listed in (1)–(5) are also computable when all spaces are replaced by their jumps. This implies that $(\mathbb{R}')_c$ is also a real algebraically closed subfield of the real numbers. Inductively, this property can be transferred to higher jumps. \square

Zheng and Weihrauch [53, Proposition 7.6] also proved that $(\mathbb{R}^{(n)})_c$ is a subfield of the reals. The fact that \mathbb{R}_c is a real algebraically closed field was proved by Rice [41] and Grzegorzczuk [23]. Freund and Staiger [20] proved that $(\mathbb{R}')_c$ is a real algebraically closed field.

Finally, we want to discuss some simple applications of our methods to differentiable functions. By $f' : [0, 1] \rightarrow \mathbb{R}$ we denote the *derivative* of a differentiable function $f : [0, 1] \rightarrow \mathbb{R}$. With very little effort we obtain the following result.

Proposition 7.5 (Operator of differentiation). *The following operation is computable: $d : \subseteq \mathcal{C}([0, 1], \mathbb{R}) \rightarrow \mathcal{C}([0, 1], \mathbb{R}')$, $f \mapsto f'$ with $\text{dom}(d) := \{f : [0, 1] \rightarrow \mathbb{R} : f \text{ differentiable}\}$.*

Proof. If $f : [0, 1] \rightarrow \mathbb{R}$ is differentiable, then

$$f'(x) = \lim_{n \rightarrow \infty} \frac{f(x + (1-x)2^{-n}) - f(x - x2^{-n})}{2^{-n}}$$

Using evaluation, currying and Theorem 5.3 we obtain that d is computable. \square

Using Theorem 5.17 we obtain the following result of Kuyper and Terwijn [30, Theorem 4.3].

Corollary 7.6 (Kuyper and Terwijn 2014). *The derivative $f' : [0, 1] \rightarrow \mathbb{R}$ of every differentiable computable function $f : [0, 1] \rightarrow \mathbb{R}$ is limit computable and continuous at all 1-generic points $x \in [0, 1]$.*

In fact, Kuyper and Terwijn obtained an even stronger result that characterizes 1-generics in terms of derivatives [30, Theorem 5.2]. Our point here is not to strengthen these results, but to illustrate the applicability of the tools that we have provided in this article.

If $f : [0, 1] \rightarrow \mathbb{R}$ is even continuously differentiable then the limit in the proof of Proposition 7.5 can be seen to be a uniform limit. This leads to the following result of von Stein [49]. By \mathcal{C}^1 we denote the set of continuously differentiable functions $f : [0, 1] \rightarrow \mathbb{R}$.

Proposition 7.7 (von Stein 1989). *$d|_{\mathcal{C}^1} : \subseteq \mathcal{C}([0, 1], \mathbb{R}) \rightarrow \mathcal{C}([0, 1], \mathbb{R}')$, $f \mapsto f'$ is computable.*

Proof. Let $f : [0, 1] \rightarrow \mathbb{R}$ be continuously differentiable. Then $F : [0, 1]^2 \rightarrow \mathbb{R}$, defined by

$$F(x, y) := \begin{cases} f'(x) - \frac{f(x) - f(y)}{x - y} & \text{if } x \neq y \\ 0 & \text{otherwise} \end{cases}$$

is continuous, and since $[0, 1]^2$ is compact, F is even uniformly continuous. Let $\varepsilon > 0$. Since F vanishes on the diagonal, there exists $\delta > 0$ such that $|x - y| < \delta \implies |F(x, y)| < \varepsilon$ for all $x, y \in [0, 1]^2$. The values $x_n := x + (1-x)2^{-n}$ and $y_n := x - x2^{-n}$ satisfy $|x_n - y_n| = 2^{-n}$. Hence there is a $k \in \mathbb{N}$ such that $|x_n - y_n| < \delta$ for all $n \geq k$. Altogether, this shows that $f_n : [0, 1] \rightarrow \mathbb{R}$ with

$$f_n(x) = \frac{f(x + (1-x)2^{-n}) - f(x - x2^{-n})}{2^{-n}}$$

satisfies $\|f_n - f'\| < \varepsilon$ for $n \geq k$. Hence, we have the uniform limit $\lim_{n \rightarrow \infty} f_n = f'$. The sequence $(f_n)_{n \in \mathbb{N}}$ in $\mathcal{C}([0, 1], \mathbb{R})$ can be computed from f using evaluation and currying. It follows that $d|_{\mathcal{C}^1}$ is computable in the given way by Theorem 5.3. \square

This result can also be phrased such that $d|_{\mathcal{C}^1} : \subseteq \mathcal{C}([0, 1], \mathbb{R}) \rightarrow \mathcal{C}([0, 1], \mathbb{R})$, $f \mapsto f'$ is limit computable. In fact, von Stein even proved that $d|_{\mathcal{C}^1}$ with this type is Weihrauch equivalent to lim , which yields for instance Myhill's result [34] that there is a continuously differentiable computable function $f : [0, 1] \rightarrow \mathbb{R}$ with a non-computable derivative $f' : [0, 1] \rightarrow \mathbb{R}$ (see [4]). Here we rather aim for the following corollary that was originally proved by Ho (announced in [25, Theorem 1.3] and proved in [26, Theorem 19]).

Corollary 7.8 (Ho 1999). *The derivative $f' : [0, 1] \rightarrow \mathbb{R}$ of every continuously differentiable computable function $f : [0, 1] \rightarrow \mathbb{R}$ is computable relative to the halting problem.*

With the following proposition we generalize the observation [10, Corollary 9.10 (2)] to all computable metric spaces.

Proposition 7.9 (C.e. comeager sets). *Let X be a computable metric space and let $(A_n)_{n \in \mathbb{N}}$ be a computable sequence of co-c.e. closed subsets $A_n \subseteq X$ that are all nowhere dense. Then $A := X \setminus \bigcup_{n \in \mathbb{N}} A_n$ is a comeager set that contains all 1-generic points $x \in X$.*

Proof. If the $A_n \subseteq X$ are nowhere dense (have non-empty interior) then $A_n = \partial A_n = \partial A_n^c$. Hence $X \setminus \bigcup_{n \in \mathbb{N}} \partial U_n \subseteq X \setminus \bigcup_{n \in \mathbb{N}} \partial A_n^c = A$. \square

We can also consider 1-generic points in the space $\mathcal{C}[0, 1]$ of continuous functions. In [5] it was proved that the set of somewhere differentiable functions is included in $\bigcup_{n=0}^{\infty} D_n$, where $(D_n)_{n \in \mathbb{N}}$ is a computable sequence of nowhere dense co-c.e. subsets $D_n \subseteq \mathcal{C}[0, 1]$, defined by $D_n := \left\{ f \in \mathcal{C}[0, 1] : (\exists t \in [0, 1])(\forall h \in \mathbb{R} \setminus \{0\}) \left| \frac{f(t+h) - f(t)}{h} \right| \leq n \right\}$. Hence, we obtain the following observation.

Corollary 7.10 (Nowhere differentiability of 1-generic functions). *Every continuous function $f : [0, 1] \rightarrow \mathbb{R}$ that is 1-generic as a point in $\mathcal{C}[0, 1]$ is nowhere differentiable.*

The mere property that every 1-generic continuous $f : [0, 1] \rightarrow \mathbb{R}$ is not differentiable could also be deduced from the fact that the operator of differentiation d is not continuous at any point f .

These examples just serve as illustrations that a careful analysis of the notions of limit computability and computability with respect to the halting problem is useful for applications in analysis. In particular the concepts based on the Galois connection between limit and Turing jumps yield very transparent and simple proofs.

REFERENCES

- [1] Spiros A. Argyros and Stevo Todorćević. *Ramsey methods in analysis*. Advanced Courses in Mathematics. CRM Barcelona. Birkhäuser Verlag, Basel, 2005.
- [2] A. V. Arkhangel'kiĭ and L. S. Pontryagin, editors. *General Topology I*, volume 17 of *Encyclopaedia of Mathematical Sciences*. Springer-Verlag, Berlin, 1990.
- [3] Vasco Brattka. *Recursive and Computable Operations over Topological Structures*. PhD thesis, Department of Computer Science, University of Hagen, Hagen, Germany, 1998.
- [4] Vasco Brattka. Computable invariance. *Theoretical Computer Science*, 210:3–20, 1999.
- [5] Vasco Brattka. Computable versions of Baire's category theorem. In Jiří Sgall, Aleš Pultr, and Petr Kolman, editors, *Mathematical Foundations of Computer Science 2001*, volume 2136 of *Lecture Notes in Computer Science*, pages 224–235, Berlin, 2001. Springer. 26th International Symposium, MFCS 2001, Mariánské Lázně, Czech Republic, August 27-31, 2001.
- [6] Vasco Brattka. Effective Borel measurability and reducibility of functions. *Mathematical Logic Quarterly*, 51(1):19–44, 2005.
- [7] Vasco Brattka, Matthew de Brecht, and Arno Pauly. Closed choice and a uniform low basis theorem. *Annals of Pure and Applied Logic*, 163:986–1008, 2012.
- [8] Vasco Brattka and Guido Gherardi. Effective choice and boundedness principles in computable analysis. *The Bulletin of Symbolic Logic*, 17(1):73–117, 2011.
- [9] Vasco Brattka, Guido Gherardi, and Alberto Marcone. The Bolzano-Weierstrass theorem is the jump of weak König's lemma. *Annals of Pure and Applied Logic*, 163:623–655, 2012.
- [10] Vasco Brattka, Matthew Hendtlass, and Alexander P. Kreuzer. On the uniform computational content of the Baire category theorem. *Notre Dame Journal of Formal Logic*, (accepted for publication), 2016.
- [11] Vasco Brattka, Matthew Hendtlass, and Alexander P. Kreuzer. On the uniform computational content of computability theory. *Theory of Computing Systems*, 61(4):1376–1426, 2017.
- [12] Vasco Brattka and Peter Hertling. Topological properties of real number representations. *Theoretical Computer Science*, 284(2):241–257, 2002.
- [13] Vasco Brattka and Gero Presser. Computability on subsets of metric spaces. *Theoretical Computer Science*, 305:43–76, 2003.

- [14] R. I. Čikvašvili. On a certain problem of Kuznecov and Trahtenbrot. *Sakharth. SSR Mecn. Akad. Moambe*, 79(2):309–312, 1975.
- [15] Matthew de Brecht. Levels of discontinuity, limit-computability, and jump operators. In Vasco Brattka, Hannes Diener, and Dieter Spreen, editors, *Logic, Computation, Hierarchies*, Ontos Mathematical Logic, pages 93–122. Walter de Gruyter, Boston, 2014.
- [16] Rodney G. Downey and Denis R. Hirschfeldt. *Algorithmic randomness and complexity*. Theory and Applications of Computability. Springer, New York, 2010.
- [17] M. Erné, J. Kosłowski, A. Melton, and G. E. Strecker. A primer on Galois connections. In *Papers on general topology and applications (Madison, WI, 1991)*, volume 704 of *Ann. New York Acad. Sci.*, pages 103–125. New York Acad. Sci., New York, 1993.
- [18] R. V. Freivald. Limiting computable functions and functionals. *Latvīšk. Gos. Univ. Učen. Zap.*, 210(Teorija Algoritmov i Programm 1):6–19, 1974.
- [19] Rudolf Freund. Real functions and numbers defined by Turing machines. *Theoretical Computer Science*, 23:287–304, 1983.
- [20] Rudolf Freund and Ludwig Staiger. Numbers defined by Turing machines. *Collegium Logicum, Annals of the Kurt-Gödel-Society*, 2:118–137, 1996.
- [21] Richard Friedberg. A criterion for completeness of degrees of unsolvability. *J. Symb. Logic*, 22:159–160, 1957.
- [22] Mark E. Gold. Limiting recursion. *The Journal of Symbolic Logic*, 30(1):28–48, 1965.
- [23] Andrzej Grzegorzczuk. Computable functionals. *Fundamenta Mathematicae*, 42:168–202, 1955.
- [24] Peter Hertling. Is the Mandelbrot set computable? *Mathematical Logic Quarterly*, 51(1):5–18, 2005.
- [25] Chun-Kuen Ho. Beyond recursive real functions. *Information and Computation*, 124(2):113–126, 1996.
- [26] Chun-Kuen Ho. Relatively recursive reals and real functions. *Theoretical Computer Science*, 210(1):99–120, 1999.
- [27] Mathieu Hoyrup. Genericity of weakly computable objects. *Theory of Computing Systems*, 60(3):396–420, 2017.
- [28] Carl G. Jockusch, Jr. Simple proofs of some theorems on high degrees of unsolvability. *Canadian Journal of Mathematics*, 29(5):1072–1080, 1977.
- [29] Alexander S. Kechris. *Classical Descriptive Set Theory*, volume 156 of *Graduate Texts in Mathematics*. Springer, Berlin, 1995.
- [30] Rutger Kuyper and Sebastiaan A. Terwijn. Effective genericity and differentiability. *J. Log. Anal.*, 6:Paper 4, 14, 2014.
- [31] A.V. Kuznécov and B.A. Trahténbrot. Investigation of partially recursive operators by means of the theory of Baire space (Russian). *Dokl. Akad. Nauk SSSR*, 105:897–900, 1955.
- [32] Alain Louveau. A separation theorem for Σ_1^1 sets. *Transactions of the American Mathematical Society*, 260(2):363–378, 1980.
- [33] Yiannis N. Moschovakis. *Descriptive Set Theory*, volume 155 of *Mathematical Surveys and Monographs*. American Mathematical Society, Providence, Rhode Island, second edition, 2009.
- [34] John Myhill. A recursive function defined on a compact interval and having a continuous derivative that is not recursive. *Michigan Math. J.*, 18:97–98, 1971.
- [35] André Nies. *Computability and Randomness*, volume 51 of *Oxford Logic Guides*. Oxford University Press, New York, 2009.
- [36] Piergiorgio Odifreddi. *Classical Recursion Theory*, volume 125 of *Studies in Logic and the Foundations of Mathematics*. North-Holland, Amsterdam, 1989.
- [37] Arno Pauly. On the topological aspects of the theory of represented spaces. *Computability*, 5(2):159–180, 2016.
- [38] Arno Pauly and Matthew de Brecht. Towards synthetic descriptive set theory: An instantiation with represented spaces. arXiv 1307.1850, 2013.
- [39] Arno Pauly and Matthew de Brecht. Non-deterministic computation and the Jayne-Rogers theorem. In Benedikt Löwe and Glynn Winskel, editors, *Proceedings 8th International Workshop on Developments in Computational Models, DCM 2012, Cambridge, United Kingdom, 17 June 2012.*, volume 143 of *Electronic Proceedings in Theoretical Computer Science*, pages 87–96, 2014.
- [40] Arno Pauly and Matthew de Brecht. Descriptive set theory in the category of represented spaces. In *30th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 438–449, 2015.
- [41] H. Gordon Rice. Recursive real numbers. *Proc. Amer. Math. Soc.*, 5:784–791, 1954.

- [42] Hartley Rogers, Jr. *Theory of Recursive Functions and Effective Computability*. MIT Press, Cambridge, second edition, 1987.
- [43] Matthias Schröder. Topological spaces allowing type 2 complexity theory. In Ker-I Ko and Klaus Weihrauch, editors, *Computability and Complexity in Analysis*, volume 190 of *Informatik Berichte*, pages 41–53. FernUniversität Hagen, September 1995. CCA Workshop, Hagen, August 19–20, 1995.
- [44] Matthias Schröder. Extended admissibility. *Theoretical Computer Science*, 284(2):519–538, 2002.
- [45] Laurent Schwartz. Sur le théorème du graphe fermé. *C. R. Acad. Sci. Paris Sér. A-B*, 263:A602–A605, 1966.
- [46] J.R. Shoenfield. On degrees of unsolvability. *Ann. of Math.*, 69(2):644–653, 1959.
- [47] Robert I. Soare. *Turing Computability. Theory and Applications of Computability*. Springer, Berlin, Heidelberg, 2016.
- [48] C. Spector. On degrees of recursive unsolvability. *Annals of Mathematics (2)*, 64:581–592, 1956.
- [49] Thorsten von Stein. *Vergleich nicht konstruktiv lösbarer Probleme in der Analysis*. PhD thesis, Fachbereich Informatik, FernUniversität Hagen, 1989. Diplomarbeit.
- [50] Klaus Wagner. Arithmetische Operatoren. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik*, 22:553–570, 1976.
- [51] Klaus Wagner. Arithmetische und Bairesche Operatoren. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik*, 23:181–191, 1977.
- [52] Klaus Weihrauch. *Computable Analysis*. Springer, Berlin, 2000.
- [53] Xizhong Zheng and Klaus Weihrauch. The arithmetical hierarchy of real numbers. *Mathematical Logic Quarterly*, 47(1):51–65, 2001.
- [54] Martin Ziegler. *Real Computability and Hypercomputation*. PhD thesis, Department of Computer Science, University of Paderborn, Paderborn, Germany, 2007. Habilitation Thesis.
- [55] Martin Ziegler. Real hypercomputation and continuity. *Theory of Computing Systems*, 41(1):177–206, 2007.
- [56] Martin Ziegler. Revising type-2 computation and degrees of discontinuity. In Douglas Cenzer, Ruth Dillhage, Tanja Grubba, and Klaus Weihrauch, editors, *Proceedings of the Third International Conference on Computability and Complexity in Analysis*, volume 167 of *Electronic Notes in Theoretical Computer Science*, pages 255–274, Amsterdam, 2007. Elsevier. CCA 2006, Gainesville, Florida, USA, November 1–5, 2006.

ACKNOWLEDGMENTS

The author would like to thank Alex Simpson for asking a question during the Logic Colloquium 2007 in Wrocław that inspired the study of the Galois connection between Turing jumps and limits. The author also acknowledges invitations by Christine Gassner and Alberto Marcone to Hiddensee and Udine, respectively, in 2017, where the results of section 3 were worked out. Last not least, a discussion with Russell Miller in Dagstuhl 2017 motivated the author to include the examples on differentiable functions, and a discussion with Arno Pauly in Oberwolfach 2018 has inspired the author to include Theorem 5.12. The author is also grateful for helpful comments by two anonymous referees that helped to improve the final version of this article.