CATEGORICAL LINEARLY ORDERED STRUCTURES

ROD DOWNEY, ALEXANDER MELNIKOV, AND KENG MENG NG

Abstract. We prove that for every computable limit ordinal α there exists a computable linear ordering $\mathcal{A}$ which is Δ^0_α-categorical and α is smallest such, but nonetheless for every isomorphic computable copy $\mathcal{B}$ of $\mathcal{A}$ there exists a $\beta < \alpha$ such that $\mathcal{A} \cong_{\Delta^0_\beta} \mathcal{B}$. This answers a question left open in the earlier work of Downey, Igusa, and Melnikov. We also show that such examples can be found among ordered abelian groups and real-closed fields.

1. Introduction

The paper contributes to the field of effective algebra [AK00, EG00], in which the main objects of study are *computable* (Rabin [Rab60]) or *constructive* (Maltsev [Mal61]) algebraic structures: A countably infinite algebraic structure $\mathcal{A}$ is computable (or constructive) if its domain can be labeled by natural numbers so that the operations and relations become Turing computable with respect to the respective labels. One of the key tensions between *computable* mathematics and classical mathematics is that the classical isomorphism type fails to preserve computable properties; what is needed for computable properties to be preserved is the notion of a *computable* isomorphism. For example, even for a structure as simple as the order type of the integers, we can show that there is an infinite collection of computable copies no pair of which are computably isomorphic.

A natural programme is to seek to understand the computational power needed to sort out an isomorphism between each pair of computable copies of a given structure. In the case that all computable copies are computably isomorphic, such as the dense linear ordering without endpoints, we say that the structure is *computably categorical*. In the case of the order type of the integers, the halting problem is enough, and necessarily so; hence this ordering is called Δ^0_2 categorical. There are obvious approaches to explore what oracles are needed to compute isomorphisms between computable structures, effective *definability* within the structures, and computational properties of the structures. Syntax, especially $\mathcal{L}_{\omega_1,\omega}$-definability, plays a rather important role in effective algebra, see [AK00]. In computable algebraic structures the syntactical complexity of their invariants usually agrees with their computational complexity.

Much work have been done on computable structures from natural algebraic structures such as groups [Khi98, EG00, Gon81], fields [EG00, Mil11], Boolean algebras [Gon97] and linear orders [Dow98]. There have also been many deep results

2010 *Mathematics Subject Classification.* Primary 03D45, 03C57. Secondary 03D75, 03D80.

The first two authors were partially supported by the Marsden Fund of New Zealand. The third author is partially supported by MOE2015-T2-2-055 and RG131/17.

This research is supported by the Mathematisches Forschunginstitut Oberwolfach through the Research in Pairs initiative in January 2018.

on abstract computable model theory [EG00, AK00]. One recent application of these results and techniques is the use of effective algebra to measure the complexity of classification problems in mathematics, most notably in group theory, see e.g. [GK02, DM08, DM14, Rig]. Also, quite unexpectedly, computable structure theory can be used to reformulate and attack the long-standing Vaught's conjecture in model theory [Mon13].

One longstanding theme in these studies has been the construction of highly pathological examples using sophisticated techniques from classical computability theory. One possible application of such techniques is in showing that certain natural subclasses of structures are unclassifiable. For example, it is well-known that linear orderings are computably categorical iff they have a finite number of successivities [Rem81]; this gives a clear classification of computable categoricity in the class of linear orders. In contrast, for *general* structures the index set of computably categorical structures (graphs) is Π^1_1-complete [DKL$^+$15], meaning that there is no hope for any reasonable invariants describing computable categoricity. Similarly, the index set of finite automata-presentable structures is Σ^1_1-complete [BHTK$^+$18]; the proof heavily exploits the techniques mentioned above. Early on, such techniques tended to apply to graphs or graphs with extended signatures of additional relations. Thus, one of the core questions in computable structure theory is the extent to which pathological results carry over to standard structures found in mathematics.

In this paper we will combine these general techniques with definability and several kinds of inversions to obtain counter-intuitive examples of computable structures in natural algebraic classes. In these examples non-uniformity occurs at the limit levels of the hyperarithmetical hierarchy, in the following sense. Answering a question of Iskander Kalimullin, in [DIM] Downey, Igusa, and Melnikov produced an example of a Δ^0_ω-categorical computable structure with ω being optimal, but such that every computable copy of the structure is $0^{(n)}$-isomorphic to the structure; recall that a computable structure $\mathcal{A}$ is Δ^0_α-categorical if any other computable $\mathcal{B} \cong \mathcal{A}$ is Δ^0_α-isomorphic to $\mathcal{A}$. (In fact, the main result of [DIM] was proved for an arbitrary computable limit ordinal.)

The structure produced in the proof of [DIM] was algebraically unnatural, i.e. it did not belong to any common algebraic class. Obviously it could be transformed into a graph, a two-step nilpotent group, or any other "universal" structure in the sense of [HKSS02, HTMMM17]. Downey, Igusa and Melnikov left open if such examples can be found in the standard non-universal classes; in particular, in the class of linear orderings. It seemed that there were serious obstacles in constructing such a linear order. However, using a careful combination of the separator technique, the label technique, and two inversions results (one of which is new), we prove:

Theorem 1.1. *For every computable limit ordinal α there exists a computable linear order $\mathcal{A}_\alpha$ such that:*

(1) *For every computable copy $\mathcal{M} \cong \mathcal{A}_\alpha$, there exists a $\beta < \alpha$ such that $\mathcal{M} \cong_{\Delta^0_\beta} \mathcal{A}_\alpha$.*
(2) *For every $\beta < \alpha$, there exists a computable copy $\mathcal{B} \cong \mathcal{A}_\alpha$ such that $\mathcal{B} \not\cong_{\Delta^0_\beta} \mathcal{A}_\alpha$.*

An example of an inversion is the map that given a Δ^0_2-linear order L, uniformly produces a computable linear order in which every point of L is replaced by an interval of order-type $\eta+2+\eta$ [DK92]. Here η denotes the order-type of the rationals. We call any such uniform procedure which de-relativizes an X-computable structure an "X-inversion". Here X can be replaced by a collection of degrees. In this paper, we view an X-inversion as a partial computable function which takes an X-index for the input to a partial computable index for the output.

We will need an extended version of the Δ^0_2-inversion for linear orders mentioned above. A second inversion which we will require is given by a well-known result of Ash [Ash86], also about linear orderings. Using the proof of our main theorem, Theorem 1.1, and the proofs of two results in the literature [Mel10, Oca14], we deduce:

Theorem 1.2. *The properties in Theorem 1.1 can be witnessed by structures from the following classes:*

(i) *Ordered abelian groups, and*
(ii) *real-closed fields of infinite transcendence degree.*

It follows from [GLS03, HTMM15] that both classes from Theorem 1.2 are not universal relative to computable dimension in the sense of [HKSS02].

We remark that both our results and those of the earlier paper give new examples of the spectrum of categoricity degrees (Fokina, Kalimullin and Miller [FKM10]). This spectrum consists of the collection of degrees which can compute an isomorphism between any two computable copies of the structure. If this collection is a principal filter in the Turing degrees, then the least element is called the degree of categoricity of the structure. For example, in the case of order type of the positive integers, this degree is $\mathbf{0}'$. In our case, we observe:

Corollary 1.3. *There exist computable structures whose categoricity spectrum is precisely the collection of upper bounds for the arithmetical degrees. Indeed, for any computable limit ordinal α, there is a structure with categoricity spectrum precisely the collection of upper bounds for degrees $\mathbf{0}^{(\beta)}$ for $\beta < \alpha$.*

Proof. Given any two copies there is some degree of the form $\mathbf{0}^{(\beta)}$ from which we can compute an isomorphism, and hence any such upper bound for such degrees can. Conversely by the construction, there is no limit to such β. □

In related work, Frolov has announced [Fro15] the existence of a computable linear ordering which is Δ^0_3-categorical but not relatively Δ^0_3-categorical.

We say a few words about the proofs of the theorems. We will adopt some of the strategies from the earlier construction [DIM] which itself was somewhat non-standard. However, Theorem 1.1 is not simply a routine coding of the structure from [DIM] into a linear order. The proof exploits several tricks, such as a simultaneous use of several inversions that are non-trivially blended with a Π^0_2-priority argument; in fact, the Π^0_2 priority construction will be performed relative to $0'''$. Thus, the reader should prepare for a technical proof.

The proof of Theorem 1.2 is similar, but it requires the use of two additional inversions, and uses a $\Pi^0_2(0^{(4)})$-priority tree. The good news is that it is very similar to the proof of Theorem 1.1, thus it will be sufficient to explain the necessary modifications to the proof of Theorem 1.1.

We remark that our paper does still leave remaining questions. For example can our result be proven for computable Boolean algebras? This would seem somewhat formidable to prove since any inversion would seem to be at level of the ω-jump.

The plan is as follows. We give a proof of Theorem 1.1 in Section 2, and we outline the proof of Theorem 1.2 in Section 3.

2. Proof of Theorem 1.1

2.1. **Notation.** For the remainder of this proof, let α be a fixed computable limit ordinal. We build $\mathcal{A} = \mathcal{A}_\alpha$. For notational convenience, when we discuss Δ^0_β constructions as oracle constructions, we will use specifically chosen Turing degree representatives that can uniformly resolve Δ^0_β questions.

Convention 2.1. Let $\langle \alpha_n : n \in \omega \rangle$ be a computable increasing sequence of ordinals whose limit is α, with $\alpha_0 > 0$. For each n, let $\beta_n = 3 + 2 \cdot \alpha_n + 1$, and note that $\langle \beta_n : n \in \omega \rangle$ is also a computable increasing sequence of ordinals whose limit is α, but with $\beta_0 > 5$. In the case when $\alpha > \omega$ we shall choose each α_n to be infinite, thus making $\beta_n = 3 + 2 \cdot \alpha_n + 1$ equal to $2 \cdot \alpha_n + 1$.

2.2. **The requirements.** Let $\langle \mathcal{M}_n : n = 1, 2, \ldots \rangle$ be a listing of all the partial computable structures in the signature of linear orders. Similarly to [DIM], we will have isomorphism requirements and diagonalization requirements:

$$\mathcal{I}_n : \mathcal{M}_n \cong \mathcal{A} \to \mathcal{M}_n \cong_{\Delta^0_{\beta_{n-1}+1}} \mathcal{A}$$

$$\mathcal{D}_n : \exists \mathcal{B}_n (\mathcal{B}_n \text{ is computable } \& \ \mathcal{B}_n \cong \mathcal{A} \ \& \ \mathcal{B}_n \not\cong_{\Delta^0_{\beta_n}} \mathcal{A})$$

Note that if we meet all of these requirements, then we will have proved the theorem. Without loss of generality, we may assume that n ranges over the positive natural numbers.

2.3. **A crude description of $\mathcal{A}$.** The linear order $\mathcal{A}$ will be a computable linear order of the form

$$\sum_{i \in \omega} 2 + \eta + 2 + P_i,$$

where each P_i will be of the form $\omega \cdot U_i$ for some non-empty U_i. Notice that every element of P_i has a successor. In the usual way, with the help of $0''$ we can locate the four extreme points in each separator $2+\eta+2$ within any (partial) computable copy of $\mathcal{A}$. Now using $0'''$ we can enumerate the first separator, the second separator, and so on, and hence $0'''$ can compute the index of the computable suborder P_i uniformly in i. Note that we are not being economical with the complexity of separators here, as we assume that all requirements have access to $0'''$.

Each such interval P_i in $\mathcal{A}$ will be controlled by a copy of some diagonalization strategy, while every isomorphism strategy will be working with co-finitely many such P_i. The exact isomorphism type of P_i will be decided dynamically in the construction. In fact, we will be using two different inversions to obtain P_i; one is well-known and is due to Ash, the other one we believe is new. Before we explain how P_i is built, we first outline the basic isomorphism strategy in a somewhat simplified context. Seeing the main idea will help to understand the role of the second inversion that deals with *labeled linear orders.*

Each diagonalization requirement will be working within an assigned interval P_n which are distinguished from other ones by separators, and it will be using a relatively powerful oracle (such as $0^{(\gamma)}$ for some $\gamma > 3$) to carry out the construction. The inversions will be applied at the end to produce the contents of P_n within the computable linear order $\mathcal{A}$. The actions of the requirement will be explained in due course. For now we only note that the diagonalization strategy is quite straightforward and does not really need "labeled" linear orderings.

2.4. **Labeled linear orderings and the basic isomorphism strategy.** The key reason we need labeled linear orderings is the strategy for the isomorphism requirement whose task is to press $\mathcal{M}_n$. We will first explain the main idea of the strategy. Then we explain how to convert labeled linear orders into usual linear orders at the cost of one additional jump.

The main idea of the pressing technique is as follows. Recall that P_i stands for a special interval of our linear ordering $\mathcal{A}$. Imagine for a moment that we allow an auxiliary c.e. unary predicate in our language of linear orders. Here c.e. means that points may satisfy the predicate, but we cannot decide it; the points satisfying the predicate can be merely enumerated. (Later we shall get rid of the predicate.) We think of the points satisfying the predicate as labeled black, and those which are not yet known to satisfy it as white. Thus, white points can turn black, but never vice versa.

Suppose at stage s our labeled linear order that we've build within P_i of $\mathcal{A}$ is:

$$\bullet \quad \bullet \quad \bullet$$

We will also attempt to build a *computable* isomorphism from $\mathcal{M}$ onto $\mathcal{A}$. At stage s we would have built a partial map $f_s : \mathcal{M}_s \to \mathcal{A}_s$. Thus, provided that $\mathcal{M}$ is actually isomorphic to $\mathcal{A}$, the interval of $\mathcal{M}$ corresponding to the location of P_i in $\mathcal{A}$ currently looks exactly the same. (For now we assume that f is able to locate the corresponding position of P_i within $\mathcal{M}$).

According to our dynamic definition of the labeled linear order in P_i which is controlled by one of the diagonalization strategies, we will attempt to extend P_i by adding one more point. We put a point but keep it white, say:

$$\bullet \quad \bullet \quad \circ \quad \bullet$$

and then wait for $\mathcal{M}$ to respond. Note that this forces $\mathcal{M}$ to give a white point at exactly the same respective location of its version of P_i. If it does not respond, or does something different, we freeze P_i in $\mathcal{A}$ and win since $\mathcal{M} \not\cong \mathcal{A}$. Indeed, $\mathcal{M}$'s version of P_i will either have too many points or have the same number of points as the P_i of $\mathcal{A}$ but is not isomorphic to it. In particular, if $\mathcal{M}$ is too quick in its enumeration then it will be "killed". As soon as we see a suitable candidate for $f^{-1}(\circ)$ (if ever), we define f on $\circ$ accordingly. Only then we make $\circ$ black:

$$\bullet \quad \bullet \quad \bullet \quad \bullet$$

and wait for the respective $\circ$ in $\mathcal{M}$ to turn black as well. If we need to extend P_i by one further point we will repeat the same strategy as described above. This finishes the description of the main idea behind the isomorphism-building strategy. (It is clear how to extend the above strategy to an arbitrary finite collection of points in P_i.)

Note that the strategy above allows us to build a *computable* isomorphism from $\mathcal{M}$ onto $\mathcal{A}$, provided we know the positions of the respective intervals controlled by the isomorphism strategies. Clearly, the biggest problem is that we do not have a c.e. predicate in our language. Also, to distinguish between various intervals P_i, P_j etc. will require a few jumps. Thus, there is more work to be done.

2.5. The two inversions used: Ash's inversion and a Δ^0_2-inversion to remove the label.

In this subsection we explain how to transform a Δ^0_2 labeled linear order into a computable linear order. Note that the labeling of the linear order is Σ^0_2.

Lemma 2.2. *There exists an effective uniform procedure Ψ that, given (an index for) a Δ^0_2 linear order L augmented with a Σ^0_2 unary predicate C, outputs (an index for) a computable linear order $\Psi(L)$ in which every x in L is replaced by $\eta + 3 + \eta$ if $C(x)$ holds, and by $\eta + 2 + \eta$ otherwise. Also, the procedure uniformly produces a Δ^0_2-map ψ that associates a point in L with the left-most point of the respective discrete interval (of size 2 or 3) in $\Psi(L)$.*

Proof of Lemma. As usual, we "densify" intervals to get rid of the wrong partial approximations. More formally, recall the standard proof in the absence of the unary predicate. Let L_s be our current best guess on the partial diagram of the Δ^0_2 order L. In L_{s+1} parts of L_s may be discovered to be wrong, e.g., what looked to the right of the point indexed by 0 will now appear to be on the left of it. To correct both the isomorphism type of the output and the embedding, "dump" the wrongly guessed intervals into the dense neighborhood of the closest currently correct-looking point in L_{s+1}. Note that the point indexed by 0 always looks correct: we know that $x \not< x$ in L. See, e.g., [DK92, Dow98] for details. Since our guess about the partial diagram of $L \restriction x$ is eventually stable for each x, this means that the positions of the "blocks" representing the points $0, \cdots, x-1$ in $\Psi(L)$ will eventually stabilize, and afterwards only grow by absorbing the errors around the block. We trust that this informal description will suffice, as this strategy is rather standard.

Now on top of the construction above, implement the following Σ^0_2 guessing procedure. If it currently looks like $C(x)$ holds on a point x in L_s, proceed in building $\eta + 3 + \eta$ instead of $\eta + 2 + \eta$ in the $\Psi(L)$-block corresponding to the point x. If at a later stage $C(x)$ no longer looks correct (which corresponds to the respective Π^0_2-predicate firing on x) we get rid of the intended right-most point in the discrete 3-interval of $\eta + 3 + \eta$ by "dumping" it into the respective future copy of η to the right of it. We then re-introduce the third (right-most) point and wait for the predicate to fire again (if ever). If the Π^0_2-predicate describing $\neg C(x)$ fires infinitely often then we are left with a copy of $\eta+2+\eta$, and we are eventually stuck with $\eta + 3 + \eta$ otherwise (unless the whole interval is destroyed due to the linear order approximation strategy outlined above).

The definition of ψ is straightforward from the construction, since at every stage we know exactly which elements of $\Psi(L)$ are representing the "2" or "3" in each block $\eta+2+\eta$ or $\eta+3+\eta$. The elementary formal details are left to the reader. $\square$

Note that the complexities Δ^0_2 and Σ^0_2 in Lemma 2.2 match the definability of the coding, and has all the nice properties of an inversion. For example, given any computable linear order C in the range of Ψ, we can uniformly build a Δ^0_2 linear order U and a Σ^0_2-labeling of U such that $\Psi(U) \cong C$. We call $U = \Psi^{-1}(C)$, so that Ψ^{-1} is also computable (from indices to indices). Also, it is obvious that $\Psi(U) \cong \Psi(\hat{U}) \Rightarrow U \cong \hat{U}$.

We will also be using the following result of Ash (Theorem 18.15 of [AK00]), about a $\Delta^0_{2\gamma+1}$-inversion operator.

Theorem 2.3 (Ash). *Let γ be a computable ordinal, and suppose L is a $\Delta^0_{2\gamma+1}$ linear ordering that does not have a least element. Then we can uniformly produce a computable presentation $F(L)$ of $\omega^\gamma \cdot L$. Moreover, we can uniformly produce a $\Delta^0_{2\gamma+1}$ function f taking $a \in L$ to the least element of the corresponding copy of ω^γ in $F(L)$.*

Similarly to Lemma 2.2, the result of Ash is sharp in the following sense. Given any computable presentation U of some $F(L)$ we can uniformly reconstruct a $\Delta^0_{2\gamma+1}$-copy S of L and find a $\Delta^0_{2\gamma+1}$ function f that plays the same role for this pair of copies (U and S) as the function does for the copies (L and $F(L)$) from the theorem. The operator F is a $\Delta^0_{2\gamma+1}$-inversion, and we define F^{-1} in a similar way.

We remark that Ash's inversion operator can be applied to β_n, for each n. In fact, we denote this operator by $F_{\Delta^0_{\beta_n}}$. Note also that $F \circ \Psi$ is defined on the domain of Ψ.

2.6. **The basic diagonalization strategy in isolation.** We describe the basic diagonalization strategy. We adopt the elementary strategy from [DIM]. Recall that $\beta_n = 3 + 2 \cdot \alpha_n + 1$.

Strategy: Build two $\Delta^0_{\beta_n}$-copies of ω^*. Here, ω^* refers to the inverse ordering on ω, where $0 > 1 > 2 > \cdots$. We diagonalize against the e^{th} potential $\Delta^0_{\beta_n}$ isomorphism $f_e : L_0 \to L_1$, as follows. Construct ω^* in both L_0 and L_1, initially letting $x_{0,e}$ and $x_{1,e}$ be adjacent elements in L_0. Wait for f_e to converge on $x_{0,e}$ and $x_{1,e}$. If the images are adjacent, insert one extra point between them, and preserve the interval.

Note that this construction is $\Delta^0_{\beta_n}$-computable. Both L_0 and L_1 have no least element, so that we can apply Theorem 2.3. In particular, we can apply the analysis from the previous subsection and – provided that the strategy is not interrupted by a higher priority strategy – will produce a pair of isomorphic computable linear orders that are not $\Delta^0_{\beta_n}$-isomorphic (recall $\beta_n > 5$). One of the two linear orders will be placed into a specific interval P_σ of $\mathcal{A}$ determined by the position of the strategy σ in the tree. The other one will be a part of a linear order $\mathcal{B}_{|\sigma|}$ which will be copying $\mathcal{A}$ everywhere except for the P_σ-interval into which the other non-$\Delta^0_{\beta_n}$-isomorphic copy of the order will be placed. The low arithmetical syntactical complexity of the separators and the rigidity of ω^{α_n} implies that $\mathcal{A} \not\cong_{\Delta^0_{\beta_n}} \mathcal{B}_n$ by the choice of $\beta_n > 5$; see the discussion of separators in Subsection 2.3.

2.7. **Putting it together: An overview of the construction.** We now discuss how exactly $\mathcal{A}$ will be built to have the desired properties, by combining all the ingredients introduced thus far. The diagonalization strategies discussed in Section 2.6 could be carried out independently of the construction and the uniformity of Ash's inversion operator will even provide us with effective sequences $\{Q^0_n\}_{n\in\omega}, \{Q^1_n\}_{n\in\omega}$ where Q^0_n, Q^1_n are $\Delta^0_{\beta_n}$-computable linear orderings which are

$\Delta^0_{\beta_n+1}$- but not $\Delta^0_{\beta_n}$-isomorphic. We might then take $\mathcal{A}$ to be perhaps something like $\sum_{n\in\omega} 2+\eta+2+F_{\Delta^0_{\beta_n}}(Q^0_n)$, where $F_{\Delta^0_{\beta_n}}$ is Ash's $\Delta^0_{\beta_n}$-inversion. We can then take each $\mathcal{B}_n$ to copy $\mathcal{A}$ everywhere except on the interval coding $F_{\Delta^0_{\beta_n}}(Q^0_n)$ where we insert $F_{\Delta^0_{\beta_n}}(Q^1_n)$ instead. This will certainly make $\mathcal{A}$ not $\Delta^0_{\beta_n}$-categorical for any n. Unfortunately, this naive amalgamation of the diagonalization outputs is too simple and does not allow us to satisfy the isomorphism $(\mathcal{I}_n)$ requirements. We shall need to combine the outputs $F_{\Delta^0_{\beta_n}}(Q^0_n)$ with the isomorphism pressing strategy discussed in Section 2.4.

The formal construction will be a standard tree argument running with oracle $0'''$. It will soon be clear why $0'''$ is necessary to run the construction, rather than having a computable construction. Suppose for the moment we accept that the formal construction has to be run with oracle $0'''$. It is therefore impossible for us to directly construct $\mathcal{A}$ as an output of the construction, for $\mathcal{A}$ has to be a computable linear order. We will instead produce $\mathcal{A}$ *indirectly* by applying the two inversions we've discussed to the $0'''$-computable objects produced by the construction.

More specifically, the construction will produce an effective sequence $\{P_n\}_{n\in\omega}$ of Δ^0_4 linear orders with Σ^0_4 labels. Some of the components P_n will be finite (when diagonalization strategies controlling an interval are abandoned or attended to finitely often), and others will be infinite (when the diagonalization strategy controlling an interval is along the true path). More information on this will be available in Section 2.8.

Since we are limiting our current discussion to a high level one, let us for now simplify our discussion and consider only one isomorphism requirement which wants to build an isomorphism from each P_n to R_n uniformly in n, where $\{R_n\}$ is some given uniform sequence of Δ^0_4 linear orders with Σ^0_4 labels, and one diagonalization requirement seeking to copy $F_{\Delta^0_{\beta_n}}(Q^0_n)$ in a component P_m. Notice that in order to meet the diagonalization requirement, it is enough to copy $F_{\Delta^0_{\beta_n}}(Q^0_n)$ inside at least one component P_m; where exactly it is copied does not really matter.

Up to now we have not yet said what the computable $\mathcal{A}$ will be; the construction will work at the Δ^0_4 level and does not directly define $\mathcal{A}$. Suppose our diagonalization requirement is of lower priority than the isomorphism requirement. It begins by enumerating the first point of $F_{\Delta^0_{\beta_n}}(Q^0_n)$ into P_m. Notice that $F_{\Delta^0_{\beta_n}}(Q^0_n)$ is computable, and since the construction works with the oracle $0'''$, we will use a Δ^0_4-index for $F_{\Delta^0_{\beta_n}}(Q^0_n)$ when copying inside P_m. At the same time we also define a Σ^0_4 labeling of P_n; for now we leave the new point of P_m white. Now the diagonalization requirement will delay enumerating anything else inside P_m and wait for the isomorphism requirement to see a corresponding (currently white) point appearing inside R_m; as R_m is uniformly Δ^0_4 with a Σ^0_4 label, this can be observed by the construction.

While waiting the diagonalization requirement will proceed to copy $F_{\Delta^0_{\beta_n}}(Q^0_n)$ inside a different fresh component $P_{m'}$, and the backup diagonalization requirement does this continuously without waiting for the isomorphism requirement to see a recovery in $R_{m'}$. If the isomorphism requirement never recovers, then P_m will be finite and the diagonalization requirement will succeed in copying $F_{\Delta^0_{\beta_n}}(Q^0_n)$ inside $P_{m'}$, and so both requirements are met. Therefore let us assume that the isomorphism requirement recovers infinitely often. In this case, each backup diagonalization attempt will be initialized every time the isomorphism requirement recovers, the backup component $P_{m'}$ will be abandoned and remain finite, and a new backup component will be picked the next time. The lead component P_m will infinitely often proceed in getting the next element of $F_{\Delta^0_{\beta_n}}(Q^0_n)$, with each new point introduced first being white, then turned black when R_m recovers. Hence $F_{\Delta^0_{\beta_n}}(Q^0_n)$ will be eventually copied inside P_m, albeit with an additional Σ^0_4 label on top of the ordering. In this case the isomorphism requirement will succeed in showing $P_m \cong_{\Delta^0_4} R_m$ as labeled orderings, and the diagonalization requirement is also satisfied in copying $F_{\Delta^0_{\beta_n}}(Q^0_n)$ inside P_m.

This concludes the description of the interactions between two requirements of different types. The full construction uses a standard tree for a Π^0_2-argument and the reader familiar with basic tree arguments will find no additional difficulty in getting all requirements to work together. We now argue that the construction at the $0'''$ level just described can be used to yield a computable linear order $\mathcal{A}$ with the desired properties.

The construction produces an effective Δ^0_4 sequence of labeled linear orders $\{P_n\}$, some of which are finite. For each fixed isomorphism requirement, it takes $0^{(5)}$ to figure out which components are infinite, and which ones are finite, and the canonical indices for the finite ones. For the components which are infinite, there are only finitely many P_m which are eventually assigned to a higher priority diagonalization requirement, and thus P_m will be $F_{\Delta^0_{\beta_n}}(Q^0_n)$ for some n; recall that each of these are $\Delta^0_{\beta_n+1}$-categorical. All the other infinite components are assigned to lower priority diagonalization requirements, and the isomorphism strategy will press on these components and build an isomorphism from these infinite components to the corresponding Q-components during the construction. Therefore, there is a large enough N such that the sequence $\{P_n\}$ is uniformly $\Delta^0_{\beta_N}$-isomorphic to $\{R_n\}$.

Now to each component P_n we first apply the relativized version of Ψ to produce a Δ^0_3 linear order $\Psi(P_n)$. We next apply Ash's Δ^0_3-inversion $F_{\Delta^0_3}$ to produce a computable linear order $F_{\Delta^0_3}(\Psi(P_n)) \cong \omega \cdot \Psi(P_n)$. This procedure is uniform in n, and so we can take

$$\mathcal{A} = \sum_{n\in\omega} 2 + \eta + 2 + F_{\Delta^0_3}(\Psi(P_n)),$$

noting that $\mathcal{A}$ is of the form promised at the beginning of Section 2.3. Now suppose that $\mathcal{M}_n \cong \mathcal{A}$. Then using $0'''$ we can uniformly recognize and enumerate the locations of the separators, and thus the corresponding intervals $\{\hat{R}_m\}$ in $\mathcal{M}_n$. We can apply the inverse of the inversions and take $R_m = \Psi^{-1}(F^{-1}_{\Delta^0_3}(\hat{R}_m))$; notice that R_m is a Δ^0_4 linear order with a Σ^0_4-labeling, and the (Δ^0_4, Σ^0_4)-indices can be found uniformly in m, n. Therefore we can run the construction with these choices of $\{R_m\}$ and hence there is a large enough N such that $\mathcal{A} \cong_{\Delta^0_{\beta_N}} \mathcal{M}_n$; if we set things up correctly we can arrange for $M = n$.

Now given n there is some component P_m where the diagonalization requirement successfully copies $F_{\Delta^0_{\beta_n}}(Q^0_n)$ inside P_m. This number m of course depends on which version of the diagonalization requirement is along the true path. Therefore the computable structure $\mathcal{B}_n$ which copies $\mathcal{A}$ everywhere except on $F_{\Delta^0_3}(\Psi(P_m))$ where it instead inserts $F_{\Delta^0_3}\left(\Psi\left(F_{\Delta^0_{\beta_n}}(Q^1_n)\right)\right)$ will not be $\Delta^0_{\beta_n}$ isomorphic to $\mathcal{A}$.

Notice that any automorphism of $\mathcal{A}$ has to map each interval $F_{\Delta^0_3}(\Psi(P_n))$ to the same corresponding interval.

2.8. **The construction.**

2.8.1. *The tree of strategies.* The tree of strategies is a standard tree for a Π^0_2-argument $\{\infty, fin\}^{<\infty}$. Each level of the tree will be monitoring the n-th partial computable structure $\mathcal{M}_n$.

Each of the two outcomes of the isomorphism-building strategy at σ will be associated with a version of the diagonalization strategy. We will call these versions the finitary and the infinitary diagonalization strategy of σ, respectively. The complexity of guessing whether $\mathcal{M}_n$ follows $\mathcal{A}$ is Π^0_2 relative to $0'''$ since it takes $0'''$ to locate the special intervals P_i in $\mathcal{M}_n$. Also, the formalization of what it means for $\mathcal{M}_n$ to follow $\mathcal{A}$ needs to be adjusted. This will be explained shortly.

Each diagonalization strategy at level n will be working relative to $\Delta^0_{\beta_n}$ (and thus has access to $0'''$) within its interval, and thus the complexity of the strategies is going up as n increases. Each such strategy can be interrupted in producing its own version of a labeled linear order, and *this interruption will be merely* $0'''$*-effective.* The two inversions will be applied to the resulting labeled linear order (finite or not) as described in Subsection 2.7.

Remark 2.4. In particular, we will *never* end up with a finite interval P_i since we will make sure the associated $0'''$-computable labeled order will always contain at least 1 point. It will then be expanded to an infinite order of the form $\omega \cdot U$ for some (infinite) U.

2.8.2. *An isomorphism-building strategy on the tree.* Each node σ at level n of the tree is given two roles: It is assigned both a diagonalization strategy as well as an isomorphism-building strategy. The outcome of a node will measure the outcome of the isomorphism-building strategy assigned to the node; the outcomes of the diagonalization strategies are not put on the tree.

The isomorphism-building strategy will attempt to build a partial isomorphism on the P-intervals (in $\mathcal{A}$ and $\mathcal{M}_n$) which are controlled by requirements of priorities weaker than that of σ. This will be done as follows. Every time the diagonalization strategy of σ, or any weaker priority diagonalization strategy below $\sigma\hat{}\infty$ puts a new point into their $0'''$-computable Σ^0_4-labeled linear order encoded into the respective interval of $\mathcal{A}$, we wait for $\mathcal{M}_n$ to respond by giving us the exact same configuration, restricted to the respective P-interval of $\mathcal{M}_n$. This process is effective in $0'''$, thus the outcome ∞ has complexity Π^0_2 *relative to* $0'''$.

There will be other versions of the diagonalization requirements played when $\sigma\hat{}fin$ is visited. This diagonalization strategy and every weaker priority diagonalization strategy below will ignore $\mathcal{M}_n$.

The notions of a σ-expansionary stage (relative to $0'''$) and the current true path at stage s are defined in the usual way.

2.8.3. *Assignment of P-intervals to the diagonalization strategies.* Recall that each such strategy is working within its own interval of the form P_i for some i (see the description of $\mathcal{A}$). If σ is at level n, then at stage s at which σ is active the first time after initialization, effectively in $0'''$ we assign P_i (with i least never used for this purpose) for the infinitary diagonalization strategy of σ. We denote this P-interval of σ by P^∞_σ.

The finitary version of the diagonalization strategy below σ will be working within its own P-interval, we denote it P_σ^{fin}. If we play $\sigma\hat{\ }fin$ and P_σ^{fin} is undefined, then effectively in $0'''$ pick P_j (where j is least never used for this purpose so far) and declare $P_\sigma^{fin} = P_j$.

2.8.4. *Initialization.* We declare P_σ^u initialized if the current true path moves to the left of $\sigma\hat{\ }u$. We set P_σ^u undefined, and we also abandon $\mathcal{B}_\sigma^u$. Note this is a $0'''$-effective process. We clarify what "abandon" means in this case. At the beginning we reserve an infinite trace of structures $C_0, C_1, \ldots$ each of which will receive $0'''$-computable instructions depending on the current true path. The inversions will be performed only on the respective P_σ^u, and the rest of $\mathcal{B}_\sigma^u$ will be *computably* copying $\mathcal{A}$. This will make the structures computable, with perhaps only one being actually isomorphic to $\mathcal{A}$. It will require $0^{(5)}$ to see which one is actually the "true" copy of $\mathcal{B}_n$, if there is any.

2.8.5. *The actions of the diagonalization strategy at expansionary stages.* Suppose $|\sigma| = n$. For each $u \in \{\infty, fin\}$, the P_σ^x-interval will attempt to implement the diagonalization strategy and also build its own version of $\mathcal{B}_n$, we denote it by $\mathcal{B}_\sigma^u$. Outside its P_σ^u-interval the structure $\mathcal{B}_\sigma^u$ will be copying $\mathcal{A}$.

Within P_σ^u, we modify the basic diagonalization strategy (Subsection 2.6) as follows. It will work effectively relative to $0'''$ (and within P_σ^u) making progress in producing its labeled $0'''$-linear order. The inversions transforming it into a computable linear order will be performed on top of that and independently of these actions. Before the strategy puts a point or changes its colour, it will wait ($0'''$-effectively) for every $\mathcal{M}_{|\tau|}$ with $\tau\hat{\ }\infty \subseteq \sigma\hat{\ }u$ to reveal the same configuration within the respective interval. Recall that the computable index of the interval P_σ^u of $\mathcal{M}_{|\tau|}$ can be uniformly reconstructed using $0'''$. Within this interval of $\mathcal{M}_{|\tau|}$, it takes $0'''$ to see the current configuration of the encoded $0'''$-linear order. The strategy will not proceed unless it gets a $0'''$-effective confirmation from each such $\mathcal{M}_{|\tau|}$. In this case we say that the stage is *σ-expansionary.*

The strategy will stop or will be paused if $\mathcal{M}_{|\tau|}$ does not respond or gives a different configuration. (For consistency, we could also apply the same pause to the respective P-interval of $\mathcal{B}_\sigma^u$, but it is not really necessary.)

2.8.6. *The construction.* Work relative to $0'''$. At stage 0, initialize all strategies. At stage s of the construction, we simply let the strategies along the current true path act according to their instructions. Independently and simultaneously, perform the jump inversions to transform the labeled $0'''$-linear orders into computable linear orders.

2.9. **Verification.** Recall that every version of the diagonalization strategy builds its own version of $\mathcal{B}_n$, and it copies $\mathcal{A}$ everywhere outside of its P-interval. We will argue that $\mathcal{M}_n \cong \mathcal{A}$ implies that the true outcome of the node σ at level n of the true path is ∞. The strategy of σ cannot control the finitely many nodes above it, but this won't be a problem since there will be an arithmetical upper bound on the complexities of isomorphisms at the exceptional intervals; the bound will be good enough to prove the theorem. For each strategy τ at deeper levels of the tree, there are only two possibilities:

Case 1. The P-interval will be eventually abandoned by the strategy. It means that the $0'''$-computable Σ_4^0-labeled linear order L produced by the strategy will be finite. After performing two inversions, we will end up with $\omega \cdot U$, where in U a black point of L is represented by $\eta + 3 + \eta$ and a white point by $\eta + 2 + \eta$. In this case the interval is uniformly $0^{(6)}$-categorical.

Case 2. Otherwise, the interval controlled by $\tau\hat{\ }u \supseteq \sigma$ will have to respect $\mathcal{M}_n$ and wait for it to respond before acting again. Therefore, all such intervals will be working towards building a $0'''$-computable isomorphism between their respective intervals in $\mathcal{M}_n$ and $\mathcal{A}$. Note that $0^{(5)}$ can compute the true path. Thus, σ can ensure that almost all P-intervals of $\mathcal{M}$ are uniformly $\Delta^0_{\beta_{n-1}+1}$-isomorphic to the respective intervals of $\mathcal{A}$.

We give the formal details. The definitions of the true path and the true outcome were standard, but relativized to $0'''$. The guessing procedure that determines the current true path is $\Pi^0_2(0''')$. The tree does not depend on our diagonalization actions whose outcomes are not even put onto the tree. It is clear that $\mathcal{M}_{|\sigma|}$ can either follow $\mathcal{A}$ or not follow $\mathcal{A}$ (in the sense of the expansionary stages), and this is exactly what each node measures.

Lemma 2.5. *For every n, the diagonalization requirement $\mathcal{D}_n$ is met within the P^x_σ-interval, where $\sigma\hat{\ }x$ lies at the true path and $|\sigma| = n$.*

Proof. For simplicity, we first consider the highest priority diagonalization requirement. According to our setup, $\sigma = e$ (the empty string) must respect $\mathcal{M}_0$. If the true outcome of $\mathcal{I}_0$ is ∞, then $\mathcal{M}_0$ always responds by copying $\mathcal{A}$ at the intermediate $0'''$-computable stage, see the previous section. Thus, the diagonalization strategy within P^∞_e will be acting at infinitely many stages. Apart from pausing at the intermediate stage, the diagonalization strategy (see 2.8.5) is no different from its simplified version described in Subsection 2.6. Recall also that the inversions are performed on the interval simultaneously with the construction. The definability complexity of the inversions lines up nicely with their effective complexity. In particular, $\mathcal{D}_0$ is met within the P^∞_e-interval.

On the other hand, if $\mathcal{M}_0$ eventually either never responds or proves to be non-isomorphic, then we implement the diagonalization strategy within some other, fresh interval P^{fin}_e which (eventually) will never be abandoned. Recall that each version of the diagonalization strategy builds its own version of $\mathcal{B}_n$ that computably copies $\mathcal{A}$ everywhere except for the P-interval controlled by the strategy (see Subsection 2.8.4). The strategy will ignore $\mathcal{M}_0$, so we will have diagonalized with the right version of $\mathcal{B}_n$.

The general case of $n > 0$ is not very much different from the basic case $n = 0$. It is sufficient to take σ with $|\sigma| = n$ along the true path and consider the interval P^x_σ, where x is the true outcome of σ. The only difference is that the strategy within P^x_σ will have to respect only those M_τ with $\tau\hat{\ }\infty \subseteq \sigma^x$. $\square$

Recall that $(\beta_n)_{n\in\mathbb{N}^+}$ has the property $0^{(\beta_n)} \geq_T 0^{(5)}$, for each n.

Lemma 2.6. *For every n, $\mathcal{I}_n$ is met.*

Proof. We need to prove that if $\mathcal{M}_n \cong \mathcal{A}$ then $\mathcal{M}_n \cong_{\Delta^0_{\beta_{n-1}+1}} \mathcal{A}$. Consider $\mathcal{M}_n$ and assume that $\mathcal{M}_n \cong \mathcal{A}$. There are several types of P-intervals that we need to consider. We argue that in each case we can (uniformly) produce an isomorphism of complexity at most $\Delta^0_{\beta_{n-1}+1}$ between the respective boxes in $\mathcal{A}$ and $\mathcal{M}_n$. Some of the analysis below was already done at the beginning of this section.

Suppose $m < n$. Then a P-interval that has ever been controlled by strategies at level m will either eventually be permanently assigned to P^x_τ for some τ of length m (which is either along or to the left of the true path), or will eventually be abandoned. With the help of $0^{(5)}$ we can see which case applies to each such an interval. It follows that $0^{(5)}$ can uniformly build an isomorphism when restricted to each such finite P-interval. By the choice of the sequence $(\beta_n)_{n\in\omega}$, $\beta_n \geq 5$. Thus, the isomorphism within each such abandoned P-interval is (uniformly) computable in $0^{(\beta_n)}$. For the finitely many $\tau\hat{\ }u$ along the true path with $|\tau| < n$, deduce from the complexity of the coding that the stable P^u_τ of $\mathcal{A}$ is uniformly relatively $\Delta^0_{\beta_{n-1}+1}$-categorical. It follows that all such P-intervals ever controlled by the higher priority strategies will be $\Delta^0_{\beta_{n-1}+1}$-categorical, non-uniformly in the finite initial segment of the true path as a parameter.

If $m \geq n$, then we appeal to the modification described in Subsection 2.8.5. Each P-interval is either eventually left finite or is stably controlled by one of the diagonalization strategies along the true path. As above, with the help of $0^{(5)}$ we can see it. If the interval is eventually abandoned then $0^{(5)}$ can produce an isomorphism. If it is never abandoned, then suppose it is permanently declared P^x_τ for some τ of length $m \geq n$. Consider σ of length n along the true path. Then σ monitors $\mathcal{M}_n$. Since $\mathcal{M}_n \cong \mathcal{A}$, it must be the case that the true outcome of σ is ∞. In particular, $\sigma\hat{\ }\infty \subseteq \tau$, for otherwise τ would have to be eventually initialized abandoning its intervals infinitely often. This means that the diagonalization strategy of P^x_τ will not add another point to its labeled $0'''$-order within its interval unless $\mathcal{M}_{|\sigma|}$ responds by giving the same configuration, see 2.8.5. It is routine to define a $0'''$-isomorphism between the labeled $0'''$-orders encoded in P^x_τ in $\mathcal{M}_{|\sigma|} = \mathcal{M}_n$ and P^x_τ in $\mathcal{A}$; see Subsection 2.4. It remains to use $0'''$ to naturally extend this map to the jump-inverted computable intervals. If the reader suspects that $0'''$ is a bit tight then they are welcomed to use $0^{(5)}$ or even $0^{(376)}$; any finite number of jumps will do.

Thus, in each case we can $\Delta^0_{\beta_{n-1}+1}$ -uniformly (in i and n) find a $\Delta^0_{\beta_{n-1}+1}$ isomorphism between P_i-boxes in $\mathcal{A}$ and $\mathcal{M}_n$. □

We conclude that all requirements are met, and thus Theorem 1.1 is proved.

3. Proof of Theorem 1.2

Melnikov [Mel, Mel10] showed that there is a uniformly effective procedure that takes a $0'$-computable linear order L into a computable ordered abelian group $G(L)$ in which the order is coded as the quotient-order by the Archimedean equivalence relation. Similarly, based on the ideas of Melnikov [Mel10], Ocasio-Gonzales [Oca14] has proved a similar result for real-closed fields. Using these results, perform the construction from the proof of Theorem 1.1 but use a $0^{(4)}$-tree of strategies and one extra inversion to pass from the linear order $\mathcal{A}$ to the respective ordered group $G(\mathcal{A})$ (or the respective real-closed field). The analysis contained in the last chapter of [Mel] and in [Oca14] shows that, given and isomorphism f between the underlying linear orders, $0'' \oplus f$ can produce an isomorphism between the respective ordered commutative structures (groups or fields). It follows that we can repeat almost literally the same analysis as in the proof of Theorem 1.1, but working relative to one extra jump.

References

[AK00] C. Ash and J. Knight. *Computable structures and the hyperarithmetical hierarchy*, volume 144 of *Studies in Logic and the Foundations of Mathematics*. North-Holland Publishing Co., Amsterdam, 2000.

[Ash86] C. Ash. Recursive labeling systems and stability of recursive structures in hyperarithmetical degrees. *Trans. Amer. Math. Soc.*, 298:497–514, 1986.

[BHTK$^+$18] Nikolay Bazhenov, Matthew Harrison-Trainor, Iskander Kalimullin, Alexander Melnikov, and Keng Meng Ng. Automatic and polynomial-time algebraic structures. preprint, 2018.

[DIM] Rod Downey, Greg Igusa, and Alexander Melnikov. On a question of kalimullin. *Proceedings of the American Mathematical Society, to appear.*

[DK92] Rodney Downey and Julia F. Knight. Orderings with αth jump degree $\mathbf{0}^{(\alpha)}$. *Proc. Amer. Math. Soc.*, 114(2):545–552, 1992.

[DKL$^+$15] Rodney G. Downey, Asher M. Kach, Steffen Lempp, Andrew E. M. Lewis-Pye, Antonio Montalbán, and Daniel D. Turetsky. The complexity of computable categoricity. *Adv. Math.*, 268:423–466, 2015.

[DM08] R. Downey and A. Montalbán. The isomorphism problem for torsion-free abelian groups is analytic complete. *J. Algebra*, 320(6):2291–2300, 2008.

[DM14] Rodney Downey and Alexander G. Melnikov. Computable completely decomposable groups. *Trans. Amer. Math. Soc.*, 366(8):4243–4266, 2014.

[Dow98] R. Downey. Computability theory and linear orderings. In *Handbook of recursive mathematics, Vol. 2*, volume 139 of *Stud. Logic Found. Math.*, pages 823–976. North-Holland, Amsterdam, 1998.

[EG00] Y. Ershov and S. Goncharov. *Constructive models*. Siberian School of Algebra and Logic. Consultants Bureau, New York, 2000.

[FKM10] Ekaterina B. Fokina, I. Kalimullin, and R. Miller. Degrees of categoricity of computable structures. *Arch. Math. Logic*, 49(1):51–67, 2010.

[Fro15] A. Frolov. Effective categoricity of computable linear orderings. *Algebra and Logic*, 54(5):415–417, 2015.

[GK02] S. Goncharov and J. Knight. Computable structure and antistructure theorems. *Algebra Logika*, 41(6):639–681, 757, 2002.

[GLS03] S. Goncharov, S. Lempp, and R. Solomon. The computable dimension of ordered abelian groups. *Adv. Math.*, 175(1):102–143, 2003.

[Gon81] S. Goncharov. Groups with a finite number of constructivizations. *Dokl. Akad. Nauk SSSR*, 256(2):269–272, 1981.

[Gon97] S. Goncharov. *Countable Boolean algebras and decidability*. Siberian School of Algebra and Logic. Consultants Bureau, New York, 1997.

[HKSS02] D. Hirschfeldt, B. Khoussainov, R. Shore, and A. Slinko. Degree spectra and computable dimensions in algebraic structures. *Ann. Pure Appl. Logic*, 115(1-3):71–113, 2002.

[HTMM15] Matthew Harrison-Trainor, Alexander Melnikov, and Antonio Montalbán. Independence in computable algebra. *J. Algebra*, 443:441–468, 2015.

[HTMMM17] Matthew Harrison-Trainor, Alexander Melnikov, Russell Miller, and Antonio Montalbán. Computable functors and effective interpretability. *J. Symb. Log.*, 82(1):77–97, 2017.

[Khi98] N. Khisamiev. Constructive abelian groups. In *Handbook of recursive mathematics, Vol. 2*, volume 139 of *Stud. Logic Found. Math.*, pages 1177–1231. North-Holland, Amsterdam, 1998.

[Mal61] A. Mal′cev. Constructive algebras. I. *Uspehi Mat. Nauk*, 16(3 (99)):3–60, 1961.

[Mel] A. Melnikov. Effective properties of completely decomposable abelian groups. *CSc dissertation (2012).*

[Mel10] A. Melnikov. Computable ordered abelian groups and fields. In *Programs, proofs, processes*, volume 6158 of *Lecture Notes in Comput. Sci.*, pages 321–330. Springer, Berlin, 2010.

[Mil11] Russell Miller. An introduction to computable model theory on groups and fields. *Groups Complexity Cryptology*, 3(1):25–45, 2011.

[Mon13] Antonio Montalbán. A computability theoretic equivalent to Vaught's conjecture. *Adv. Math.*, 235:56–73, 2013.

[Oca14] Victor Ocasio González. *Computability in the Class of Real Closed Fields*. PhD thesis, Notre Dame University, 2014.

[Rab60] M. Rabin. Computable algebra, general theory and theory of computable fields. *Trans. Amer. Math. Soc.*, 95:341–360, 1960.

[Rem81] J. B. Remmel. Recursively categorical linear orderings. *Proc. Amer. Math. Soc.*, 83(2):387–391, 1981.

[Rig] K. Riggs. The decomposability problem for torsion-free abelian groups is analytic complete. Proceedings of the American Mathematical Society (to appear).

Victoria University of Wellington
Email address: `Rod.Downey@msor.vuw.ac.nz`

Massey University
Email address: `alexander.g.melnikov@gmail.com`

Nanyang Technological University
Email address: `kmng@ntu.edu.sg`