Computable Cyclic Functions

Ross Horne

May 18, 2005

Abstract

This dissertation concerns computable analysis where the idea of a representations of a set is of central importance. The key ideas introduced are those commenting on the computable relationship between two newly constructed representations $\lambda^1_{\vartheta,\rho^2}$, a representation of integrable cyclic functions, and $\left[\vartheta \to \rho^2\right]$, the continuous cyclic function representation. Also the computable relationship of an absolutely convergent Fourier series representation, α_{ρ^2}, is considered. It is observed that $\lambda^1_{\vartheta,\rho^2}$ gives rise to a much larger set of computable functions than obtained by $\left[\vartheta \to \rho^2\right]$ and that integration remains a computable operation but that basic evaluation of the function is not computable. Many other representations are acknowledged enhancing the picture of the partial order structure on the space of representations of cyclic functions. The paper can also be seen as a foundation for the study of Fourier analysis in a computable universe and concludes with an investigation into the computability of the Fourier transform.

Contents

1 Introduction

This paper aims to investigate the result of restricting the mathematical theory of analysis to a computable world. This aim is further enhanced in two ways. Firstly a non-trivial domain of analysis in which to perform the investigation is considered. The domain chosen will be Fourier analysis which gives a huge scope to work over and has the added benefit of leaving a wide and accessible open end for further development of the ideas. The investigation will stem from the foundations and cover the basic building blocks of the theory giving a thorough evaluation of the computability structure.

The second way in which the aim of the project will be enhanced is through the choice of a non-trivial theory of computability with which to perform the investigation. The theory in question, Type-2 Theory of Effectivity (TTE), is laid out elegantly in [1] by Klaus Weihrauch of the University of Hagen in Germany and is the result of three decades of his own study and the work of other mathematicians such as Hauck, Grezegorczyk and, of course, Turing.

There are several reasons for selecting TTE from the huge range of theories available for studying computability in analysis. The first is the concept of a representation which this project will focus on. Informally a representation is a way of interpreting the inputs and output of a computation in an abstract mathematical universe so, depending upon the representation, one computation can have several abstract mathematical meanings or several computations can have the same abstract mathematical meaning. This is not an alien concept but is side lined as being trivial in most computability theories which is fair enough since for finite computations, which computer science generally deals with, the theory gained by including representations is trivial. However in computable analysis, where generally infinite inputs and outputs will be dealt with, the theory of representations is far from trivial. The second reason for selecting TTE is its relative youth leaving open vast unsaturated areas of fresh theory to be investigated. The third but not least temptation of TTE is its elegance, due mainly to work done in [1]. Simple uncluttered foundations, which in many ways mimic the foundations of topology, give rise to a neat logical algebra which in turn facilitates neat concise proofs. Theorem 8.2 is an example of a result which would most likely take up several pages in other computability theories to convey the same ideas.

Naturally the fundamental concepts of the theory will be conveyed in sufficient detail for the needs of the paper. This will allow the construction of the first building block which will be required for the study of Fourier analysis — the one dimensional torus group, $\mathbb{T}$. Although the real numbers have already been developed in TTE and could be used to build the space of cyclic functions through restriction of the domain or the function set itself, neither are satisfactory solutions. On the other hand $\mathbb{T}$ provides a neat way of conveying and indeed forcing all the required properties of the cyclic functions which are to be considered. A variety of representations of $\mathbb{T}$ will be considered and compared also giving some initial insight into the significance of the theory itself.

The paper will continue by studying representations of functions of the form $f : \mathbb{T} \to \mathbb{C}$. For this, in addition to the representations of $\mathbb{T}$, a variety representations of $\mathbb{C}$ will be required although less work is required here since representations of $\mathbb{C}$ can be constructed logically from representations of $\mathbb{R}$ and $\mathbb{T}$. From representations of $\mathbb{T}$ and $\mathbb{C}$ various representations of continuous computable functions will be constructed which will be a natural initial choice for a set of functions to consider given the close relationship between computability and topology which will feature throughout the paper. Following a brief outline of the theory of sequence spaces under TTE, representations of integrable functions will be considered and the investigation will conclude with consideration to a space of particularly well behaved functions, namely those with absolutely convergent Fourier series. Relevant examples of computable operations such as integration will be considered in these represented function spaces at most stages including multiplication, integration and the Fourier transform.

It is hoped that, for the reader, the paper itself will be palatable and the significance of the results obtained in the paper will be appealing on three levels. The first being on the purely mathematical level since many results are interesting in themselves, with both an elegant computability theory and an elegant mathematical theory being the subject of scrutiny. The second level to consider would be the aesthetic appearance of the partial order structure on the space of representations of cyclic functions under computable reduction although only a glimpse of this transfinite lattice will be experienced. Finally there are the practical applications of these results which increase the number of computable functions and provide the maximal space of functions that one can operate in given that the operations, which are desired to be performed computably, are known.

As a rough guide to the reader it is worth considering the distribution of material. The beginning of the paper is largely a rephrasing and consolidation of back ground material and will be most useful as a reference for specific aspects of notation used. The first section of original material is the section on $\mathbb{T}$ although some standard lemmas are incorporated. Around the middle of the paper there is a section on standard string function representations which is indeed a standard part of the theory although results and definition are, where ever possible, generalised further. From that point onwards all definitions are original so certainly all results are not only original but of heightened interest.

2 Fundamental concepts

This section will build up the concept of strings and their abstraction functions called naming systems. Strings are considered to be concrete objects which by themselves have no meaning in abstract mathematics. However, a naming system is a partial function which identifies strings with a unique abstract object. Such abstraction functions will also give meaning to string functions in particular those realised by TTE which will also be defined in this section.

2.1 Strings

The following definitions regarding strings are included for completeness and as an explanations of notation used in proofs later on. Here it is most important to note and consider the topologies chosen for the sets of strings.

Definition 2.1. *A string is a list of characters from a fixed finite alphabet Σ. The set of all finite strings is denoted Σ^* and the set of all countably infinite strings is denoted Σ^ω. The topology on Σ^* will be the discrete topology, τ_*, and on Σ^ω the Cantor topology, τ_C,*

where $\tau_C := \{A\Sigma^\omega | A \subseteq \Sigma^*\}$

where $A\Sigma^\omega$ is the set of all infinite strings beginning with A.

The following notation will be useful for referring to strings and extracting the information encoded in them.

1. *Repeated concatenation is denoted $\otimes_{i=1}^{n+1} a_i := (\otimes_{i=1}^{n} a_i) a_n$ with base case $\otimes_{i=1}^{1} a_i := a_1$ where $a_i \in \Sigma$. Note that the concatenation operator is omitted for strings and simply denoted by juxtaposition which is in line with how string are commonly presented. Also this idea can be extended for infinite strings by taking limits of initial segments for example.*

2. *Let $\iota : \Sigma^* \to \Sigma^*$ such that $0, 1 \in \Sigma$ and $\iota(\oplus_{i=1}^{n} a_i) = 110(\oplus_{i=1}^{n} a_i 0)11$ where $\forall i, a_i \in \Sigma$. This will be used for wrapping up elements within a longer string so they do not interfere with each other.*

3. *Let $u \in \Sigma^*$ and $p \in \Sigma^a$ where $a \in \{*, \omega\}$. $u \sqsubseteq p$ iff u is an initial segment of p, i.e. there exists $q \in \Sigma^a$ such that $uq = p$.*

4. *Let $u \in \Sigma^*$ and $p \in \Sigma^a$ where $a \in \{*, \omega\}$. $u \lhd p$ iff u is a contiguous subword of p, i.e. there exists $v \in \Sigma^*$ and $q \in \Sigma^a$ such that $vuq = p$.*

5. *Let $\langle u, w\rangle := \iota(u)\iota(w)$ where $u, w \in \Sigma^*$, $\langle u, p\rangle := \iota(u)p$ where $u \in \Sigma^*$ and $p \in \Sigma^\omega$ and $\langle p, q\rangle := \otimes_{i=1}^{\infty} p_i q_i$ where $p, q \in \Sigma^\omega$ and $p = \otimes_{i=1}^{\infty} p_i, q = \otimes_{i=1}^{\infty} q_i$. The pairings' inverse projections are denoted $\langle . \rangle_1$ and $\langle . \rangle_2$.*

   N.B. Pairings can be extended to n dimensions by nesting and to infinite dimensions by the Cantor pairing function.

Note that $\Sigma = \{\mathtt{0}, \mathtt{1}\}$ suffices for the purpose of representation since a fixed number of characters can be encoded much like the ASCII notation but would lead to bloated definitions and proofs. It is therefore reasonable to assume that the alphabet used will be implicit from the situation. Also the basic operations above will be assumed to be computable in the sense soon to be established.

2.2 Type-2 theory of effectivity

The TTE model of computation can be used to model computers with infinite inputs and outputs. A *type-2 machine*, M, is a Turing machine consisting of input tapes labelled 1 to k, a finite number of work tapes and an output tape. Operation is as with a normal Turing machine except all the Turing machine can do on the input tape is read a character on the tape and move along to the next symbol and similarly on the output tape can only write a character and move along to the next space. In addition, M includes a *type specification* $(a_i)_{i=0}^k$, where $\forall i, a_i \in \{*, \omega\}$, since only one of either infinite or finite strings may be handled on an input/output tape.

Each type-2 machine realises a *computable string partial function* f_M in the following manner. Let $(x_i)_{i=1}^k \in \prod_{i=1}^k \Sigma^{a_i}$ and for $i = 1$ to k, place the characters for x_i on tape i starting at the head and extending to the right. If $a_i = *$ then complete the remaining tape with the character $\mathtt{B}$. The first case is $a_0 = *$, in which case $f_M\,(x_i)_{i=1}^k := y \in \Sigma^*$ iff M halts on input $(x_i)_{i=1}^k$ with y on the output tape. Otherwise, $a_0 = \omega$ in which case $f_M\,(x_i)_{i=1}^k := y \in \Sigma^\omega$ iff M writes each initial segment of y to the output tape in finite time on input $(x_i)_{i=1}^k$.

So, a string partial function $f : \prod_{i=1}^n \Sigma^{a_i} \to \Sigma^{a_0}$ is computable iff $f = f_M$ for some type-2 machine M. Also a string $p \in \Sigma^\omega$ is computable iff $p = f(\mathtt{0})$ for some computable string partial function $f : \Sigma^* \to \Sigma^\omega$. Note that every $u \in \Sigma^*$ is computable under a similar definition.

Here we will assume that Church's Thesis holds to a sufficient degree so that careful algorithmic descriptions in English suffice.

2.3 Naming systems

There are two types of *naming systems* which can be conceived of under the physical limitations of our own existence. The majority of results in this paper will concern naming systems. They can be thought of as defining the meanings of strings which machines can manipulate.

Definition 2.2. *1. A notation of an abstract mathematical object A is an onto partial function $\nu : \Sigma^* \to A$.*

2. A representation of an abstract mathematical object A is an onto partial function $\delta : \Sigma^\omega \to A$.

2.4 Equivalence of naming systems

Similarly to topological equivalence two naming systems, γ, γ', are equivalent, denoted $\gamma \equiv \gamma'$, iff there is a computable function translating both ways between the two naming systems so one naming system can be computably reduced to the other.

Definition 2.3. *Let $\gamma : \Sigma^a \to M$ and $\gamma' : \Sigma^{a'} \to M'$ be naming systems.*

1. *$\gamma \leq \gamma'$ iff there exists a computable function $f : \Sigma^a \to \Sigma^{a'}$ such that for each $y \in dom(\gamma)$, $f(y) \in dom(\gamma')$ and $\gamma(y) = \gamma' f(y)$. f is said to translate γ to γ'.*
2. *$\gamma \equiv \gamma' \Leftrightarrow \gamma \leq \gamma' \wedge \gamma' \leq \gamma$.*
3. *$\leq_t$ and $\equiv_t$ are defined by replacing computable with continuous in the above respective definitions.*

2.5 Computable abstract functions

This key idea essentially allows computability of abstract functions to be characterised by the computability of underlying string functions with respect to specific naming systems. The relationship between concepts in the definition for the case $n = 1$ is aided be the figure below.

Definition 2.4. *Let $(\gamma_i : \Sigma^{a_i} \to M_i)_{i=0}^n$ be naming systems of abstract mathematical objects $(M_i)_{i=0}^n$ respectively where $a_i \in \{*, \omega\}$. An abstract partial function $f : \prod_{i=1}^n M_i \to M_0$ is $((\gamma_i)_{i=1}^n, \gamma_0)$-computable iff there exists a computable partial string function $g : \prod_{i=1}^n \Sigma^{a_i} \to \Sigma^{a_0}$ such that $\gamma_0 \circ g = f \circ (\gamma_i)_{i=0}^n$.*

For a definition of $((\gamma_i)_{i=1}^n, \gamma_0)$-continuous abstract functions simply replace computable with continuous in the definition above.

$$\begin{tikzcd}
\Sigma^{a_1} \arrow[r, "g"] \arrow[d, "\gamma_1"'] & \Sigma^{a_0} \arrow[d, "\gamma_0"] \\
M_1 \arrow[r, "f"'] & M_0
\end{tikzcd}$$

2.6 Further Concepts

The computability of individual objects is of less importance in this paper but is worth referring to when considering the motivation and consequences of results.

Definition 2.5. *Let $\gamma : \Sigma^a \to M$ be a naming system of M where $a \in \{*, \omega\}$. An object $y \in M$ is γ-computable iff $y = \gamma(p)$ where $p \in \Sigma^a$ is computable.*

Several computability and topological concepts on subsets of an abstract object can be defined with respect to a naming system in a standard way. Examples include whether a subset of M is $\gamma - open$ or $\gamma - r.e.$ and make use of the fact that *open* and *r.e.* sets are well defined on the underlying sets of strings. The following definition schema demonstrates how this is done in its most general form.

Definition 2.6. *For $i = 1$ to n let $\gamma_i : \Sigma^{a_i} \to M_i$ where $a_i \in \{*, \omega\}$ and let $H_i \subseteq M_i$. $\prod_{i=1}^n H_i$ is $(\gamma_i)_{i=1}^n$-property iff $\prod_{i=1}^n \gamma_i^{-1}(H_i)$ is property with respect to $\prod_{i=1}^n \Sigma^{a_i}$.*

2.7 The continuity of computability

The following result ties together the two fields of continuity and computability. The result is well known in several computability models but is central to the results in this paper and can be elegantly proven in TTE as shown in a rephrasing from [1] and can be converted into a more useful form for comparing naming systems as shown in the corollory.

Theorem 2.7. *A computable partial function $f : \prod_{i=1}^n \Sigma^{a_i} \to \Sigma^{a_0}$ is continuous where $a_i \in \{*, \omega\}$ and the product topology is as commonly defined.*

Proof. Suppose that $a_0 = \omega$. Let $w \in \Sigma^*$ such that $f(p) \in w\Sigma^\omega$ where $p \in dom(f)$. By the computability of f, there exists a machine M such that $f = f_M$ which outputs w, an initial segment of $f(p)$, in finitely many steps, hence only using the finite initial segment, $u \in (\Sigma^*)^n$, of p. Thus $\forall p' \in$

$u(\prod_{i=1}^{n}\Sigma^{a_i}) \cap dom(f)$, $f(p') \in w\Sigma^\omega$ so $u(\prod_{i=1}^{n}\Sigma^{a_i}) \cap dom(f) \subseteq f^{-1}(w\Sigma^\omega)$. Finally, $\{w\Sigma^\omega | w \in \Sigma^*\}$ is a base for τ_C and so the above is sufficient to prove the continuity of f.

Otherwise $a_0 = *$. Let $w \in \Sigma^*$ such that $f(p) = w$ where $p \in dom(f)$. By the computability of f, there exists a machine M such that $f = f_M$ which outputs w in finitely many steps then halts, hence only using the finite initial segment, $u \in (\Sigma^*)^n$, of p. Thus $\forall p' \in u(\prod_{i=1}^{n}\Sigma^{a_i}) \cap dom(f)$, $f(p') = w$ so $u(\prod_{i=1}^{n}\Sigma^{a_i}) \cap dom(f) \subseteq f^{-1}(\{w\})$. Finally, $\{\{w\}|w \in \Sigma^*\}$ is a base for τ_* and so the above is sufficient to prove the continuity of f. $\square$

Corollary 2.8. *If γ, γ' are naming systems and $\gamma \leq \gamma'$, then $\gamma \leq_t \gamma'$.*

Proof. Consider $\gamma : \Sigma^a \to M$ and $\gamma : \Sigma^{a'} \to M'$ where $a, a' \in \{*, \omega\}$. Suppose that $\gamma \leq \gamma'$ then there exists a computable string function $f : \Sigma^a \to \Sigma^{a'}$ such that for all $p \in dom(\gamma), f(p) \in dom(\gamma')$ and $\gamma(p) = \gamma'(f(p))$. By the above theorem, f is also continuous, hence $\gamma \leq_t \gamma'$. $\square$

2.8 Philosophy behind TTE

As a brief aside note that this model is not only useful for studying computability but also models the way in which the human being operates where concrete objects are processed in the brain and abstract objects are those which we are observing. Strings can be seen as those things we can store or write down. Only a finite amount of information about an observation of the world around us can ever be assimilated even if hypothetically there were an infinite amount of information to assimilate. Infinite strings can be seen as a sequence of recordings of observations. A naming system is the association between what was recorded and observed. Those objects observed may be mathematical and it is clear that in set theory alone there are more than countably many objects. Since there are only countably many finite strings it is clear that not all objects can be accessed by one notation or even countably many representations. For instance, abstract sets could be represented through strings which are logical deductions from the ZF axioms. However, an application of the axiom of choice will not in general yield a unique object so cannot be included in the representation. Such objects that cannot be accessed without the axiom of choice are the non-computable sets under this representation. Hence only countably many abstract sets can be specifically accessed in this manner. This is the key motivation for this area of study since both $\mathbb{T}$ and $\mathbb{C}$ have 2^ω elements and $\mathbb{C}^\mathbb{T}$ has 2^{2^ω} elements hence "almost all" their elements are non-computable.

3 Representations of $\mathbb{R}$

The representations of $\mathbb{R}$ are of secondary importance and are covered extensively in [1] Chapter 4 so will not be covered here in detail. However a few results will be useful to ease results on representations of $\mathbb{T}$ and $\mathbb{C}$ which will be cited along the way.

For now regarding the partial order structure on the representations of $\mathbb{R}$ it will be accepted that [1] defines a representation called ρ, which will be referred to later, and several other representations, which for simplicity will be restricted to two representations $\rho_\perp$ and $\rho_\top$ where $\rho_\perp < \rho < \rho_\top$ i.e. there is a computable reduction only in one direction between these representations. Also, assume that there is a representation $\rho_C \equiv \rho$ which is defined in a similar manner to ϑ_C which appears in the following section.

It is interesting to note, at this stage to get an idea of the point in considering different representations, that the most widely spread and familiar representation of the reals, i.e. an infinite sequence of digits in base n with a place holder, does not have some nice properties which will be encountered later and also leaves undesirable results for example multiplication by three is not computable (or even continuous).

4 Representations of $\mathbb{T}$

The one dimensional torus group or circle group $\mathbb{T}$ can be defined to be the quotient group $\mathbb{R}/2\pi\mathbb{Z}$ with group operator $+$. The natural topology on $\mathbb{T}$, say $\tau_\mathbb{T}$, can be formed by letting $p : \mathbb{R} \to \mathbb{T}$ such that $p(x) := \dot{x} = x + 2\pi\mathbb{Z}$, i.e. projecting x onto its coset, and letting $V \in \tau_\mathbb{T} \Leftrightarrow V = p[U]$ where U is open in $\mathbb{R}$. See [6] 2.1.1 for more details on $\mathbb{T}$.

There are many ways in which $\mathbb{T}$ can be represented, some naturally yielding subtly but significantly different topologies. This section will name a few and show how they relate computably.

4.1 Effective topological spaces

Effective topological spaces defined below give rise to equivalence classes of naming systems. These admissible naming systems, defined below, preserve the topological structure of the effective topological space, also defined below.

Definition 4.1. *1. An effective topological space is a triple $\boldsymbol{S} = (M, \sigma, \nu)$ where σ is a countable set of subsets of M such that*

$$(\forall x, y \in M)\, x = y \Leftarrow \{U \in \sigma | x \in U\} = \{U \in \sigma | y \in U\}$$

and ν is a notation of σ.

2. *$\tau_{\boldsymbol{S}}$ is the topology on M with subbase σ.*

3. *A computable topological space is an effective topological space such that $\{(u,v)|u,v \in dom(\nu) \wedge \nu(u) = \nu(v)\}$ is r.e..*

4. *A standard representation, $\delta_{\boldsymbol{S}}$, of an effective topological space, $\boldsymbol{S} = (M, \sigma, \nu)$ is defined such that*

$$(p \in dom(\delta_{\boldsymbol{S}}) \wedge \iota(w) \lhd p) \Rightarrow w \in dom(\nu)$$

$$\delta_{\boldsymbol{S}}(p) = x \Leftrightarrow \{A \in \sigma | x \in A\} = \{\nu(w)|\iota(w) \lhd p\}$$

where $w \in \Sigma^*$, $x \in M$ and $p \in \Sigma^\omega$.

Definition 4.2. *A naming system, γ, is admissible with respect to τ iff $\gamma \equiv_t \delta_{\boldsymbol{S}}$ for some effective topological space $\boldsymbol{S}$ with $\tau = \tau_{\boldsymbol{S}}$.*

The significance of admissible naming systems comes through in the following "Main Theorem" from [1] 3.2.11 which essentially says that the continuity of an abstract functions is dependent on the continuity of the underlying concrete string partial function when considered with respect to admissible representations. It is interesting to note that continuity is also preserved in some inadmissible representation hence admissibility is a sufficient but not necessary condition as shown with the naive Cauchy representation on real numbers in [3].

Theorem 4.3. *Let $(\delta_i)_{i=0}^n$ be admissible naming systems with respect to topologies $(\tau_i)_{i=0}^n$ on sets $(M_i)_{i=0}^n$ respectively. Then for any partial function $f : \prod_{i=1}^n M_i \to M_0$, f is $((\tau_i)_{i=1}^n, \tau_0)$-continuous iff f is $((\delta_i)_{i=0}^n, \delta_0)$-continuous.*

4.2 Admissible notations of $\mathbb{N}$ and $\mathbb{Q}$

To begin the construction of an admissible representation of $\mathbb{T}$ it is necessary to consider first the basic underlying sets. However the specific construction of the notations below are unimportant since not only has a lot of work already been done on such notations but also the topological concepts arising from notations are trivial as suggested by the theorem below which shows that the notations defined in this section are trivially admissible. None the less it is worth checking that they exist and developing a notation for them. This highlights why naming systems have been side lined in other computability theories since they would generally be concerned with notations only, although in complexity theory they would have more of an impact.

Definition 4.4. *1. Let $\nu_{\mathbb{N}}$ be a notation of $\mathbb{N}$ such that $dom(\nu_{\mathbb{N}}) = \{0, 1\}^*$ and $\nu_{\mathbb{N}}\left((a_i)_{i=0}^{n}\right) = \sum_{i=0}^{n} a_{n-i} \cdot 2^i$.*

2. Let $\nu_{\mathbb{Z}}$ be a notation of $\mathbb{Z}$ such that $\nu_{\mathbb{Z}}(w) = \nu_{\mathbb{N}}(w)$ and $\nu_{\mathbb{Z}}(\text{-}w) = -\nu_{\mathbb{Z}}(w)$ where $w \in dom(\nu_{\mathbb{N}})$.

3. Let $\nu_{\mathbb{Q}}$ be a notation of $\mathbb{Q}$ such that $\nu_{\mathbb{Q}}(w/u) = \nu_{\mathbb{Z}}(w)/\nu_{\mathbb{N}}(u)$ where $\nu_{\mathbb{N}}(u) \neq 0$.

Theorem 4.5. *A notation ν of M is admissible w.r.t. τ iff τ is the discrete topology on M. See [1] 3.2.8.4.*

4.3 Admissible representations of $\mathbb{T}$

The topological concepts arising from representations are less trivial. Here the key admissible representation of $\mathbb{T}$ is defined.

Definition 4.6. *1. Let $\|\dot{x}\| := \min\{|x + 2\pi n| : n \in \mathbb{Z}\}$.*

2. Let $Cb := \{B(a,r) | a \in \mathbb{Q}, r \in \mathbb{Q}, r > 0\}$ where

$$B(a,r) := \{y \in \mathbb{T} : \|y - \dot{a}\| < r\}.$$

3. Let I be a notation of Cb where

$$I(\langle v, w\rangle) := B(\nu_{\mathbb{Q}}(v), \nu_{\mathbb{Q}}(w)).$$

Definition 4.7. *Let $\boldsymbol{S}_{=} := (\mathbb{T}, Cb, I)$.*

The following result is a slight tangent as the computability of an effective topological is not used in any later results. It has been included since it is an original result and may be of use in further study in this field.

Theorem 4.8. *$\boldsymbol{S}_{=}$ is a computable topological space.*

Proof. Let $\dot{x}, \dot{y} \in \mathbb{T}$ and suppose that assume $X = Y$, where

$$Y := \{U \in Cb : \dot{y} \in U\}$$

$$X := \{U \in Cb : \dot{x} \in U\}$$

and let $n \in \mathbb{N}$. Since $\mathbb{Q}$ is dense in $\mathbb{R}$, there exists $a_n \in \mathbb{Q}$ such that $\|\dot{x} - \dot{a_n}\| < 1/(2n)$. Hence $B(a_n, 1/(2n)) \in X \Rightarrow B(a_n, 1/(2n)) \in Y \Rightarrow \|\dot{y} - \dot{a_n}\| < 1/(2n)$. Therefore $\|\dot{x} - \dot{y}\| = \|\dot{x} - \dot{a_n} + \dot{a_n} - \dot{y}\| \leq \|\dot{x} - \dot{a_n}\| +$

$\|\dot{a_n} - \dot{y}\| < 1/n$, by the triangle inequality. Therefore $\|\dot{x} - \dot{y}\| = 0$ so $\dot{x} = \dot{y}$. So $\mathbf{S}_=$ is an effective topological space since also I is a notation of Cb.

$\mathbf{S}_=$ is computable since a suitable type-2 machine M which takes inputs $\langle v_1, w_1\rangle$, $\langle v_2, w_2\rangle \in dom(I)$ can be defined. Since each coset of a rational number only contains one rational number which is conveniently the number it is characterised by, it is sufficient for the machine to check that each v_i and w_i is in $dom(\nu_\mathbb{Q})$ and that either $\nu_\mathbb{Q}(v_1) = \nu_\mathbb{Q}(v_2)$ and $\nu_\mathbb{Q}(w_1) = \nu_\mathbb{Q}(w_2)$ hold or $\nu_\mathbb{Q}(w_1) > \pi$ and $\nu_\mathbb{Q}(w_2) > \pi$ hold at which point M halts in an accept state if all checks hold and diverges otherwise. □

Definition 4.9. *Let ϑ be the standard representation of $\boldsymbol{S}_=$.*

Lemma 4.10. *ϑ is admissible w.r.t. $\tau_\mathbb{T}$.*

Proof. It is sufficient to show that $\tau_\mathbb{T} = \tau_{\mathbf{S}_=}$ which follows from the observation that if $B(a, r) \in Cb$ and $D(a, r) := \{x \in \mathbb{R} : |x - a| < r\}$ then $B(a, r) = p(D(a, r))$, hence a base for $\tau_{\mathbf{S}_=}$ is contained in $\tau_\mathbb{T}$, so $\tau_{\mathbf{S}_=} \subseteq \tau_\mathbb{T}$. Conversely, for every positive real, R, there exists an increasing sequence of positive rational numbers $(r_i)_{i=0}^\infty$ with limit R so $\bigcup_{i=0}^\infty B(a, r_i) = p(\bigcup_{i=0}^\infty D(a, r_i)) = p(D(a, R))$. $\{D(a, R) : a, R \in \mathbb{R}\}$ is well known to be a basis for the topology of $\mathbb{R}$ hence $\tau_\mathbb{T} \subseteq \tau_{\mathbf{S}_=}$. □

4.4 The representations $\vartheta_<$ and $\vartheta_>$

There are other topologies which can be placed on $\mathbb{T}$ with their associated admissible naming systems which give rise to different sets of computable and continuous functions. Two of these are characterized by the computable topological spaces listed below. The spaces are computable by similar arguments to those on $\mathbf{S}_=$ so will not be replicated here.

Definition 4.11. *Let the following computable topological spaces be defined.*

- $\boldsymbol{S}_< := (\mathbb{T}, \sigma_<, \nu_<)$ *where*

$$\begin{aligned} N &:= \max\{n \in \mathbb{Z} : 2\pi n \le \nu_\mathbb{Q}(w)\} \\ \nu_<(w) &:= \{\dot{x} : 2\pi N \le x < \nu_\mathbb{Q}(w)\} \\ \sigma_< &= range(\nu_<). \end{aligned}$$

- $\boldsymbol{S}_> := (\mathbb{T}, \sigma_>, \nu_>)$ *where*

$$\begin{aligned} N &:= \min\{n \in \mathbb{Z} : \nu_\mathbb{Q}(w) \le 2\pi n\} \\ \nu_<(w) &:= \{\dot{x} : \nu_\mathbb{Q}(w) < x \le 2\pi N\} \\ \sigma_< &= range(\nu_>). \end{aligned}$$

Also, let $\vartheta_<, \vartheta_>$ be the standard representations of $\mathbf{S}_<, \mathbf{S}_>$.

4.5 Relationship between $\vartheta_<$, $\vartheta_>$ and ϑ

The first two results here are examples of computability results obtained by "cheating" and looking at continuity first. Combined with the other results in this section a small section of the partial order structure on the representations of $\mathbb{T}$ is obtained including a neat way they are logically connected through conjunction.

Lemma 4.12. *If $\gamma : \Sigma^a \to M, \gamma' : \Sigma^{a'} \to M$ are naming systems of M where $a, a' \in \{*, \omega\}$, U is γ-open and $\gamma' \leq_t \gamma$, then U is γ'-open.*

Proof. Let U be γ-open. Since $\gamma' \leq_t \gamma$, there exists a continuous $f : \Sigma^a \to \Sigma^{a'}$ such that $\gamma'(p) = \gamma(f(p))$ for all $p \in dom(\gamma')$. By definition $\gamma^{-1}[U]$ is open in $dom(\gamma)$ and, by continuity of f, $(f^{-1} \circ \gamma^{-1})[U]$ is open in $dom(\gamma')$ hence $U = ((\gamma \circ f) \circ (\gamma \circ f)^{-1})[U] = (\gamma' \circ (f^{-1} \circ \gamma^{-1}))[U] \Leftrightarrow (\gamma')^{-1}[U] = (f^{-1} \circ \gamma^{-1})[U]$ so U is γ'-open . □

Lemma 4.13. *Neither $\vartheta_< \leq_t \vartheta$, nor $\vartheta_> \leq_t \vartheta$, nor $\vartheta_< \leq \vartheta$, nor $\vartheta_> \leq \vartheta$ hold.*

Proof. Suppose that $\vartheta_< \leq_t \vartheta$, then $p(-1, 1)$ is ϑ-open but is clearly not $\vartheta_<$-open contradicting Lemma 4.12. Now, suppose that $\vartheta_< \leq \vartheta$, then by corollary 2.8, $\vartheta_< \leq_t \vartheta$ contradicting the first part of the proof. An identical proof works for $\vartheta_>$. □

Definition 4.14. *If γ, γ' are naming systems, then $\gamma \wedge \gamma' : \langle p, q \rangle \mapsto x \Leftrightarrow \gamma(p) = \gamma'(q) = x$ is a naming system.*

Lemma 4.15. *If $\gamma, \gamma_1, \gamma_2$ are naming systems then $\gamma \leq \gamma_1 \wedge \gamma \leq \gamma_2 \Leftrightarrow \gamma \leq (\gamma_1 \wedge \gamma_2)$.*

Proof. Suppose that $\gamma \leq \gamma_1 \wedge \gamma \leq \gamma_2$ so there exist computable functions f_1, f_2 such that $\gamma(p) = \gamma_1(f_1(p))$ and $\gamma(p) = \gamma_1(f_1(p))$ where $p \in dom(\gamma)$. So $\gamma_1(f_1(p)) = \gamma_2(f_2(p)) = \gamma(p)$, hence $(\gamma_1 \wedge \gamma_2) \langle f_1(p), f_2(p) \rangle = \gamma(p)$ and $p \mapsto \langle f_1(p), f_2(p) \rangle$ is clearly computable since composition preserves computability. Therefore $\gamma \leq (\gamma_1 \wedge \gamma_2)$.

Conversely, if $\gamma \leq (\gamma_1 \wedge \gamma_2)$ then there exists a computable function f such that $(\gamma_1 \wedge \gamma_2) f(p) = \gamma(p)$ where $p \in dom(\gamma)$. So $f(p) = \langle p_1, p_2 \rangle$ and $\gamma_1(p_1) = \gamma_2(p_2) = \gamma(p)$ and also $\langle f \rangle_1, \langle f \rangle_2$ are computable, since composition preserves computability, hence $\gamma \leq \gamma_1 \wedge \gamma \leq \gamma_2$. □

Theorem 4.16. *$\vartheta \leq \vartheta_<$, $\vartheta \leq \vartheta_>$ and $(\vartheta_< \wedge \vartheta_>) \equiv \vartheta$.*

Proof. Let $p \in dom(\vartheta)$. There exists a type-2 machine which takes each $\iota(\langle v, w \rangle) \lhd p$ and outputs a list of all the resulting $\iota(u)$ where $\nu_{\mathbb{Q}}(u) =$

$\nu_{\mathbb{Q}}(v) + \nu_{\mathbb{Q}}(w)$ hence $\vartheta \leq \vartheta_{<}$. Similarly, if $\nu_{\mathbb{Q}}(v) + \nu_{\mathbb{Q}}(w)$ is replaced with $\nu_{\mathbb{Q}}(v) - \nu_{\mathbb{Q}}(w)$ in the above then $\vartheta \leq \vartheta_{>}$ holds. Hence, by the above lemma, $\vartheta \leq \vartheta_{<} \wedge \vartheta_{>}$. Conversely, if $\langle p, q\rangle \in dom(\vartheta_{<} \wedge \vartheta_{>})$ then there exists a type-2 machine, M, which considers each $\iota(v_i) \lhd p$ and $\iota(w_i) \lhd q$, where $i \in \mathbb{N}$, under the computable function $P : (v, w) \mapsto \iota(\langle v', w'\rangle)$ where $2\nu_{\mathbb{Q}}(v') = \nu_{\mathbb{Q}}(w) + \nu_{\mathbb{Q}}(v)$ and $2\nu_{\mathbb{Q}}(w') = \nu_{\mathbb{Q}}(w) - \nu_{\mathbb{Q}}(v)$. Clearly the list $r := \oplus_{i=1}^{\infty} \left(\oplus_{j=1}^{i-1} P(v_i, w_j)\, P(v_j, w_i)\right) P(v_i, w_i)$ can then be output by M. Furthermore $\vartheta(r) = \vartheta_{<} \wedge \vartheta_{>} \langle p, q\rangle$ hence $\vartheta \leq (\vartheta_{<} \wedge \vartheta_{>})$. $\square$

4.6 The Cauchy representation

Theorem 4.16 is a starting point for showing that the representation ϑ is not unique as a representation of $\mathbb{T}$ admissible with respect to the standard topology in fact there are infinitely many equivalent representations. The Cauchy representation is one such representation itself having several equivalent forms. It will also be more practical for showing the computability of function with an argument represented by ϑ which can be done since equivalent representations can be exchanged freely.

Definition 4.17. *Let the naive Cauchy representation, ϑ_{Cn}, of $\mathbb{T}$ be defined such that $\vartheta_{Cn}(p) = x$ iff $p = \otimes_{i=1}^{\infty}\iota(w_i)$ where $\forall i, w_i \in dom(\nu_{\mathbb{Q}})$, $\nu_{\mathbb{Q}}(w_i) = x_i$ and also $x = \lim_{i\to\infty} \dot{x}_i$.*

Definition 4.18. *Let the Cauchy representation, ϑ_C, be the same as the naive Cauchy representations except that in addition*

$$\forall m, n > N, \|\dot{x_m} - \dot{x_n}\| < 2^{-N}.$$

Theorem 4.19. $\vartheta \equiv \vartheta_C$

Proof. Consider the type-2 machine M which takes input $\oplus_{i=1}^{\infty}\iota(w_i) \in dom(\vartheta_C)$ and generates the output $\oplus_{\langle i,j\rangle=1}^{\infty} v_{i,j}$ where $\vartheta_C\left(\oplus_{i=1}^{\infty}\iota(w_i)\right) = \dot{x}$ and

$$v_{i,j} = \begin{cases} u_j & \text{if } u_j \in dom(I) \text{ and } B\left(\dot{\nu_{\mathbb{Q}}}\left(w_i\right); 2^{-i}\right) \subseteq I(u_j) \\ u_{\perp} & \text{otherwise} \end{cases}$$

where u_j is the j^{th} string in a fixed enumeration of Σ^* and $u_\perp$ is such that $I(u_\perp) = \mathbb{T}$.

Since $\dot{x} \in I(u_j)$ iff there exist an i such that $B\left(\dot{\nu_{\mathbb{Q}}}\left(w_i\right); 2^{-i}\right) \subseteq I(u_j)$, every ball containing x will eventually be included so $\vartheta\left(\oplus_{\langle i,j\rangle=1}^{\infty} v_{i,j}\right) = x$ hence $\vartheta_C \leq \vartheta$. Note that the decidability of $B\left(\dot{\nu_{\mathbb{Q}}}\left(w_i\right); 2^{-i}\right) \subseteq I(u_j)$ is a consequence of Lemma 4.23.

Conversely, consider a type-2 machine, M', which takes input $p \in dom(\vartheta)$ and works in stages. At stage 0 it simply searches for $\iota\left(\langle w_0, v_0\rangle\right) \lhd p$ such that $\nu_{\mathbb{Q}}(v_0) < 1$ and writes $\iota(w_0)$ to the output tape. At stage $i+1$ it sequentially considers each $\iota(u) \lhd p$ until one is found such that $I(u) \subseteq B\left(\dot{\nu_{\mathbb{Q}}}\left(w_i\right); 2^{-i}\right)$. Consider $u = \langle w_{i+1}, v\rangle$ and extend the output tape with $\iota(w_{i+1})$ before moving to stage $i+2$. Hence $\vartheta_C\left(f_{M'}\left(p\right)\right) = \vartheta(p)$ so $\vartheta \leq \vartheta_C$.

□

It is interesting to note that for $\mathbb{T}$ more work has been done than necessary since the set of cosets of natural numbers are dense in $\mathbb{T}$ so would be sufficient as the elements of the sequence encapsulated by ϑ_C rather than the set of cosets of $\mathbb{Q}$. However this is not important since the representations would be intertranslatable and hence equivalent; but is interesting to note that is an element of neatness $\mathbb{R}$ does not posses.

The next results look at the relationship of the naive Cauchy representation compared to other the other representations thus far developed.

Definition 4.20. *If γ_1, γ_2 are naming systems, then $\gamma_1 \vee \gamma_2$ is defined such that $\texttt{01}p \mapsto \gamma_1(p)$ and $\texttt{001}p \mapsto \gamma_2(p)$ is a naming system.*

Lemma 4.21. *If $\gamma, \gamma_1, \gamma_2$ are naming systems then $\gamma_1 \leq \gamma \wedge \gamma_2 \leq \gamma \Leftrightarrow (\gamma_1 \vee \gamma_2) \leq \gamma$.*

Proof. Suppose that $\gamma_1 \leq \gamma$ and $\gamma_2 \leq \gamma$ so there exists computable functions f_1, f_2 such that $\gamma_1(p_1) = \gamma(f_1(p_1))$ and $\gamma_2(p_2) = \gamma(f_1(p_2))$ where $p_1, p_2 \in dom(\gamma_1), dom(\gamma_2)$ respectively. Let M be a type-2 machine which, on input $p \in dom(\gamma_1 \vee \gamma_2)$ check whether it begins with $\texttt{01}, \texttt{001}$ upon which point it executes f_1, f_2 respectively on the remainder of p. Hence f_M translates $(\gamma_1 \vee \gamma_2)$ to γ.

Conversely, if $(\gamma_1 \vee \gamma_2) \leq \gamma$, then there exists a computable function f such that $(\gamma_1 \wedge \gamma_2)(p) = \gamma(f(p))$ where $p \in dom(\gamma_1 \vee \gamma_2)$. Both $g_1 : p \mapsto \texttt{01}p$ and $g_2 : p \mapsto \texttt{001}$ are clearly computable and when $p \in dom(\gamma_1)$ $\gamma_1(p) = (\gamma \circ f \circ g_1)(p)$ hence $\gamma_1 \leq \gamma$, since composition preserves computability. Similarly $\gamma_2 \leq \gamma$. □

Theorem 4.22. *$\vartheta_< \leq \vartheta_{Cn}$, $\vartheta_> \leq \vartheta_{Cn}$.*

Proof. Suppose $\vartheta_<(p) = x$. So there exists a type-2 machine, M, which searches for the first $\iota(w_0) \lhd p$ such that $w_0 \in dom(\nu_{\mathbb{Q}})$ and writes $\iota(w_0)$ to the output tape; there after at each stage, i, it considers w_i which has most recently been written to the output tape and considers each $u \lhd p$ until it finds one such that $\Lambda\left(\dot{0}, \dot{\nu_{\mathbb{Q}}}(w_i)\right) \geq \Lambda\left(\dot{0}, \dot{\nu_{\mathbb{Q}}}(u)\right)$ which is computable by Lemma 4.23 since equality is easily checked for rationals and trivially $\dot{\nu_{\mathbb{Z}}} \leq \vartheta_C$.

A similar proof works for $\vartheta_>$. □

Hence, by lemma 4.21, $(\vartheta_< \vee \vartheta_>) \leq \vartheta_{Cn}$ However, unlike conjunction, disjunction does not give rise to the inverse result i.e. $(\vartheta_< \vee \vartheta_>) \geq \vartheta_{Cn}$ since nothing about $x = \vartheta_{Cn}(p)$ can be deduced from an initial segment of p. Hence ϑ_{Cn} induces the trivial topology on $\mathbb{T}$ so is only interesting as a top element in the topological reduction hierarchy.

Here is the computable reduction partial order on $\mathbb{T}$ so far.

$$\begin{tikzcd}
& \vartheta_{Cn} & \\
& \vartheta_> \vee \vartheta_< \arrow[u] & \\
\vartheta_> \arrow[ur] & & \vartheta_< \arrow[ul] \\
& \vartheta \arrow[ul] \arrow[ur] &
\end{tikzcd}$$

4.7 Some computable functions on $\mathbb{T}$

Inequalities between elements of $\mathbb{T}$ are not expressible in the same way as on real numbers hence positioning of elements must be compared by other means. An abstract image of $\mathbb{T}$ which the representations give rise to can be a circle which increases in one direction or the other, say anti-clockwise keeping in line with a well known projection into $\mathbb{C}$ — the mapping $\dot{x} \mapsto e^{xi}$. This will be used in the following function which has an important precondition i.e. that two identical elements are not being considered, a condition which is computationally impossible to verify one of the critical reasons for considering $\mathbb{T}$ rather than a restriction of $\mathbb{R}$ to $[0, 2\pi)$.

Lemma 4.23. *$\Lambda : \mathbb{T}^2 \to \mathbb{R}$, such that $\Lambda(\dot{x}, \dot{y})$ = the anti-clockwise distance between $\dot{x}$ and $\dot{y}$ where $\dot{x} \neq \dot{y}$, is $(\vartheta, \vartheta, \rho)$-computable.*

Proof. Let $\vartheta_C(p) = \dot{x}$ and $\vartheta_C(q) = \dot{y}$. Let a type-2 machine, M, take inputs p, q and work as follows: At stage i it searches for the $(i+2)^{th}$ $\iota(u) \lhd p$ such that $u \in dom(\nu_{\mathbb{Q}})$ and the $i + 2^{th}$ $\iota(v) \lhd q$ such that $v \in dom(\nu_{\mathbb{Q}})$. By the definition of ϑ_C, $\|u - x\| \leq 2^{-(i+2)}$ and $\|v - y\| \leq 2^{-(i+2)}$ hence, by the triangle inequality on $\|.\|$,

$$\|(v - u) - (y - x)\| \leq \|u - x\| + \|y - v\| \leq 2^{-(i+1)}$$

. Also trivially, by classical computability theory $(v, u) \mapsto w := v - u$ is $(\nu_{\mathbb{Q}}, \nu_{\mathbb{Q}}, \nu_{\mathbb{Q}})$-computable.

By [Wei2000 Example 4.3.13.8] 2π is ρ''_C-computable i.e. can be compatibly generated $\oplus_{i=1}^{n}\iota(\varpi_i)$ such that $|\nu_{\mathbb{Q}}(\varpi_i) - 2\pi| < 2^{-i}$.

Consider w:

If $w = 0$ then output nothing and begin M at stage $i + 1$.

If $w > 0$ then work in stages beginning at 0 as follows: At stage j consider $m_i := w - j\varpi_{i+j+1}$ which is $(\nu_{\mathbb{Q}})$-computable. If $0 \leq m_i < \varpi_{i+j+1} + 2^{-(i+1)}$ then go to the next section, otherwise continue to stage $j + 1$.

If $w < 0$ then work in stages beginning at 0 as follows: At stage j consider $m_i := w + j\varpi_{i+j+1}$ which is $(\nu_{\mathbb{Q}})$-computable. If $0 \leq m < \varpi_{i+j+1} + 2^{-(i+1)}$ then go to the next section, otherwise continue to stage $j + 1$.

Both above processes will terminate since

$$\begin{aligned} |(w \pm j\varpi_{i+j+1}) - (w \pm j2\pi)| &= |j(2\pi - \varpi_{i+j+1})| \\ &= |j|\,|(2\pi - \varpi_{i+j+1})| \\ &\leq |j|\,2^{-(i+j+1)} \\ &\leq 2^{-(i+1)} \end{aligned}$$

hence will pass within the specified termination range.

Now consider whether $m_i < 2^{-i}$ or $\varpi_{i+2} - 2^{-i} < m_i$. If so then write nothing to the output tape and begin M to stage $i + 1$. Otherwise extend the output with $\iota(m)$.

Suppose $\dot{x} \neq \dot{y}$. Since $\mathbb{T}$ is a Hausdorff space, there exists N such that $\|m_i\| \geq 2^{-i}$ for all $i \geq N$ so the first case above will occur only finitely many times and it doesn't matter that some steps may have output nothing in the process since the proximity to $\Lambda(\dot{x}, \dot{y})$ will be at least within the desired precision for the representation ρ_C. Note that $\dot{x} = \dot{y}$ cannot be decided so it can never be told in finite time whether $\Lambda(\dot{x}, \dot{y})$ will in fact converge to some value within 2^{-i} of 2π rather than of 0 if $\Lambda(\dot{x}, \dot{y})$ and 0 cannot be distinguish between at stage i.

Since $\vartheta_C \equiv \vartheta$ and $\rho_C \equiv \rho$ the proposition holds. □

Lemma 4.24. *The set* $\{(x, z, y) \subseteq \mathbb{T}^3 :$ *z is anti-clockwise strictly between x and y* $\}$*, is* $(\vartheta, \vartheta, \vartheta)$*-r.e..*

Proof. Let $f(x, y, z) = \mathtt{1} \Leftrightarrow \Lambda(x, y) < \Lambda(x, z)$. □

Also of use in this project for constructing the group characters for use in the Fourier transform will be the multiplication of an element of $\mathbb{T}$ by an integer. To establish the computability of multiplication first the computability of addition on $\mathbb{T}$ must be considered. It is interesting to note that the idea of extending multiplication to the product of two elements of

$\mathbb{T}$ ran into difficulties in particular because of the question of the existence of such an operation. For instance, is $1/2 \dot{\cdot} \dot{0}$ equal to $\dot{0}$ or $\dot{\pi}$ or even say $4\pi^2 \dot{+} \pi$, highlighting another major difference between $\mathbb{R}$ and $\mathbb{T}$. This also shows that what has been notated as a norm on $\mathbb{T}$ is not in fact a norm but merely defines a metric since if it were a norm then the computability of "multiplication" on $\mathbb{T}^2$ could be proven by pointwise rational multiplication of the elements of the Cauchy sequence, but beginning four elements in to the sequence, which contradicts the fact that a suitable multiplication does not exist.

Theorem 4.25. *$(x, y) \mapsto x + y$ is $(\vartheta, \vartheta, \vartheta)$-computable.*

Proof. It is well known, by classical computability theory, that $(x, y) \mapsto x + y$ is $(\nu_{\mathbb{Q}}, \nu_{\mathbb{Q}}, \nu_{\mathbb{Q}})$-computable so there is a type-2 machine M which transforms inputs $p, q \in dom(\vartheta_C)$ where $p = \oplus_{i=0}^{\infty}\iota(u_i)$ and $q = \oplus_{i=0}^{\infty}\iota(v_i)$ to $r = \oplus_{i=0}^{\infty}\iota(w_i)$ where $\nu_{\mathbb{Q}}(w_i) = \nu_{\mathbb{Q}}(u_{i+1}) + \nu_{\mathbb{Q}}(v_{i+1})$. Now for all i

$$\begin{aligned}
\|w_i - (x + y)\| &= \|\nu_{\mathbb{Q}}(u_{i+1}) + \nu_{\mathbb{Q}}(v_{i+1}) - (x + y)\| \\
&\leq \|\nu_{\mathbb{Q}}(u_{i+1}) - x\| + \|\nu_{\mathbb{Q}}(u_{i+1}) - x\| \\
&\leq 2^{-i-1} + 2^{-i-1} \\
&\leq 2^{-i}.
\end{aligned}$$

So $r \in dom(\vartheta_C)$ and $\vartheta_C(r) = x + y$ hence f_M realises the $(\vartheta_C, \vartheta_C, \vartheta_C)$-computability of $(x, y) \mapsto x + y$ hence by theorem 4.19 the result holds. □

Theorem 4.26. *$(n, \theta) \mapsto n \cdot \theta$ is $(\nu_{\mathbb{Z}}, \vartheta, \vartheta)$-computable.*

Proof. Firstly consider when $n \geq 0$ and let $H : \mathbb{N} \times \mathbb{T} \to \mathbb{T}$ such that

$$\begin{aligned}
H(0, \theta) &= \theta \\
H(n + 1, \theta) &= H(n, \theta) + \theta.
\end{aligned}$$

H is $(\nu_{\mathbb{N}}, \vartheta, \vartheta)$-computable by [1] 3.1.7.3 (iteration) since by Theorem 4.25 addition is $(\vartheta, \vartheta, \vartheta)$-computable. However when $n < 0$ Let $H(n, \theta) = -H(-n, \theta)$ which is well defined since, given a type-2 machine, M, transforming rationals in the Cauchy representation to their negative, f_M is a $(\vartheta_C, \vartheta_C)$-realisation of negation on $\mathbb{T}$. The result follows by theorem 4.19 and the preservation of computability through composition. □

5 Representations of $\mathbb{C}$

The representations of $\mathbb{R}$ and $\mathbb{T}$ can be used to construct a huge selection of distinct representations of $\mathbb{C}$ simply by pairing naming systems together to create a new one.

Definition 5.1. *Let $[\gamma_1, \gamma_2]$ be a naming system of $M_1 \times M_2$, where γ_1, γ_2 are naming systems of M_1, M_2 respectively, such that*

$$[\gamma_1, \gamma_2] \langle p_1, p_2 \rangle := (\gamma_1 (p_1), \gamma_2 (p_2)).$$

Definition 5.2. *Let $[\gamma_1, \gamma_2]$ be the representation of $\mathbb{C}$ such that γ_1, γ_2 are representations of $\mathbb{R}$ and $(x, y) = [\rho_1, \rho_2](p) \Rightarrow x + iy = [\rho_1, \rho_2](p)$.*

Definition 5.3. *Let $[\rho_1, \vartheta_2]$ be the representation of $\mathbb{C}$ such that ρ_1 is a representation of $\mathbb{R}$, ϑ_2 is a representation of $\mathbb{T}$ and $(x, \dot{y}) = [\rho_1, \vartheta_2](p) \Rightarrow |x| e^{iy} = [\rho_1, \vartheta_2](p)$.*

5.1 Admissible representations of $\mathbb{C}$

Admissibility on such naming systems can be derived from one result below showing that $[\rho, \rho]$, abbreviated to ρ^2, is admissible since in [1] 4.1.3 ρ is shown to be an admissible representation of the real numbers. Similarly $[\rho, \vartheta]$ is admissible.

Lemma 5.4. *Suppose that γ_1, γ_2 are admissible naming systems of M_1, M_2 with respect to τ_1, τ_2 respectively. Then $[\gamma_1, \gamma_2]$ is admissible with respect to the product topology $\tau_1 \otimes \tau_2$.*

Proof. For $i = 1, 2$, since γ_i is admissible with respect to τ_i, there exist computable topological spaces $\mathbf{S}_i = (M_i, \sigma_i, \nu_i)$ such that $\tau_{\mathbf{S}_i} = \tau_i$ and $\gamma_i \equiv_t \delta_{\mathbf{S}_i}$. Let $\mathbf{S} = (M_1 \times M_2, \sigma, \nu)$ where $\nu(\texttt{01}w) := \nu_1 \times M_2$ and $\nu(\texttt{001}w) := M_1 \times \nu_2$. Hence $\sigma = range(\nu)$ is a base for $\tau_1 \otimes \tau_2$ so $\tau_1 \otimes \tau_2 = \tau_{\mathbf{S}}$.

Now, let a type-2 machine, M, be defined such that given input $\langle p_1, p_2 \rangle \in dom([\delta_{\mathbf{S}_1}, \delta_{\mathbf{S}_2}])$, at stage i, with q_i on the output tape, it alternates between p_1, p_2 looking for the next $\iota(u_1) \lhd p_1, \iota(u_2) \lhd p_2$ not yet observed then extend the output tape to $q_i \iota(\texttt{01}u_1)\iota(\texttt{001}u_2)$. Hence f_M translates $[\delta_{\mathbf{S}_1}, \delta_{\mathbf{S}_2}]$ to $\delta_{\mathbf{S}}$.

Conversely assume type-2 machine M', with work tapes buffer 1 and buffer 2, takes input $p \in dom(\delta_{\mathbf{S}})$ at stage i looks for the next $\iota(\texttt{01}w_1)$, $\iota(\texttt{001}w_2) \lhd p$ not yet observed and extends buffer 1 with $\iota(w_1)$ and buffer 2 with $\iota(w_2)$. Finally it extends the output tape by removing the first character from buffer 1 and placing it on the output tape followed by the first of buffer 2. Hence $f_M$translates $\delta_{\mathbf{S}}$ to $[\delta_{\mathbf{S}_1}, \delta_{\mathbf{S}_2}]$.

Therefore $\delta_{\mathbf{S}} \equiv [\delta_{\mathbf{S}_1}, \delta_{\mathbf{S}_2}] \Rightarrow \delta_{\mathbf{S}} \equiv_t [\delta_{\mathbf{S}_1}, \delta_{\mathbf{S}_2}]$ by Corollary 2.8. $\square$

Theorem 5.5. $[\rho, \vartheta] \equiv [\rho, \rho]$

This is evident from establishing the (ϑ, ρ)-computability of sin and cos which in turn would show the (ϑ, ρ^2)-computability of exp. Conveniently, exactly the same string function used to realise that sin and cos are (ρ_C, ρ_C)-computable as provided in [1] Example 4.3.13 will realise their (ϑ_C, ρ_C)-computability. This is because of the 2π periodicity of sin which means that a rational number taken from a remote part of the real line can still produce the required precision of proximity to the result.

6 Representations of continuous functions

Now that representations of both $\mathbb{C}$ and $\mathbb{T}$ have been obtained representations of the functions of interest — $f : \mathbb{T} \to \mathbb{C}$ — can be built. To begin with, notations for the set of computable string functions will be developed and from those a class of representations of continuous string functions will be obtained and used as a tool to represent the desired cyclic functions.

6.1 The notation of computable string functions

Definition 6.1. *Let $a, b, c \in \{*, \omega\}$. Let G^{ab} be a set of functions $g : \Sigma^a \to \Sigma^b$ and let $\zeta : \Sigma^c \to G^{ab}$ be a naming system of G^{ab}.*

$utm(\zeta)$: There exists a computable partial function $u : \Sigma^c \times \Sigma^a \to \Sigma^b$ such that $\zeta_x(y) = u(x, y)$ for all $x \in dom(\zeta)$ and $y \in \Sigma^a$.

$s^m_n(\zeta)$: For any computable partial function $f : \Sigma^c \times \Sigma^a \to \Sigma^b$ there exists a computable total function $s : \Sigma^c \to \Sigma^c$ such that $f(x, y) = \zeta_{s(x)}(y)$ for all $x \in \Sigma^c$ and $y \in \Sigma^a$.

Definition 6.2. *Let $a, b \in \{*, \omega\}$. Let ξ^{ab} be a notation of the set of computable partial functions $f : \Sigma^a \to \Sigma^b$, say P^{ab}, such that $utm(\xi^{ab})$ and $s^m_n(\xi^{ab})$ hold.*

Lemma 6.3. *Any notation for F^{ab} satisfying utm and s^m_n are equivalent.*

Proof. Let δ, γ be naming systems of F^{ab} satisfying *utm* and s^m_n. By $utm(\delta)$, there exists a computable partial function $g : \Sigma^* \times \Sigma^a \to \Sigma^b$ such that $g(x, y) = \delta_x(y)$ and so, by $s^m_n(\gamma)$, there exists a computable partial function $s : \Sigma^* \to \Sigma^*$ such that $g(x, y) = \gamma_{s(x)}(y)$. Therefore s translates δ to γ, since $\delta_x(y) = g(x, y) = \gamma_{s(x)}(y)$, and by symmetry of argument there exists an s' translating γ to δ. □

The above lemma verifies the validity of the above definition since it is arbitrary which ξ^{ab} is chosen, for a fixed a, b, and eliminates a lengthy definition of ξ^{ab} classically constructed in this situation. As a slight tangent, but a worth while one and one worthy of study in its own right, this definition can be extended to higher orders of computability, say computable function acting on sets of characters of size ω, 2^ω, etc, by allowing the computable string function be those notated by any non-deterministically chosen naming system satisfying an extended version of the *utm* and s_n^m properties. Essentially this has been done here with sets of size ω in the following section but any further generalisation is not required for this paper and is of a still more abstract nature.

6.2 The standard representations of continuous string functions

Here $\xi^{\omega b}$ is used to construct a representation of continuous functions which will be seen to be very natural. Lemma 6.5 proves the validity of the representation. See [1] 2.3.11 for the full result although the case $a = b = \omega$, which will be used most in this paper since both $\mathbb{T}$ and $\mathbb{C}$ require infinite strings to be represented, is outlined here.

Definition 6.4. 1. *Let F^{*b} be the set of partial functions $f : \Sigma^* \longrightarrow \Sigma^b$.*

Let $F^{\omega *}$ be the set of partial functions $f : \Sigma^\omega \longrightarrow \Sigma^*$ which are continuous and dom(f) is open.

Let $F^{\omega\omega}$ be the set of partial functions $f : \Sigma^\omega \longrightarrow \Sigma^\omega$ which are continuous and dom(f) is a G_δ set.

A G_δ set is the intersection of a sequence of open sets.

2. *The standard representation of F^{ab}, $\eta^{ab} : \Sigma^\omega \longrightarrow F^{ab}$, is*

$$\eta^{ab}(\langle x, p\rangle)(y) := \xi_x^{\omega b} \langle p, y\rangle$$

for all $x \in \Sigma^*$, $p \in \Sigma^\omega$ and $y \in \Sigma^a$.

Lemma 6.5. *η^{ab} is a representation of F^{ab}.*

Proof. Case $a = b = \omega$: Let f be in *range*$(\eta^{\omega\omega})$ i.e. Let $x \in \Sigma^\omega$ and $p \in \Sigma^\omega$ and consider $\eta^{\omega\omega}_{\langle x,p\rangle}$. For all $q \in \Sigma^\omega$, $g(q) := \langle p, q\rangle$ is computable hence continuous and $\eta^{\omega\omega}_{\langle x,p\rangle}(q) = \xi_x^{\omega\omega} \langle p, q\rangle = \xi_x^{\omega\omega} \circ g(q)$. Since $\xi_x^{\omega\omega}$ is computable hence continuous and the composition of continuous functions is continuous, $f \in F$.

Conversely, let $f \in F$ so there exits an $h : \Sigma^* \to \Sigma^*$ such that $f(y) = \sup\{z \in \Sigma^* : \exists n \in \mathbb{N} \text{ s.t. } h(y_{<n}) = z\}$ where $y_{<n}$ is the first n characters of y. $\{\langle y, z\rangle : h(y) = z\}$ can be listed as $p \in \Sigma^\omega$ (p is essentially the infinite program). Define a Type-2 Machine, M, which on input $\langle p, q\rangle$ at stage n searches for a pair $\langle y, z\rangle$ in the string p such that y is an initial segment of q and z_n is an initial segment of z where z_n is what was on the output tape at the start of stage n. Before going into stage $n + 1$ it writes the remaining characters in z to the output tape. So M realises f_M and there exists an x such that $\xi^{\omega\omega}_x = f_M$. $\eta^{\omega\omega}_{\langle x,p\rangle}(p) = \xi^{\omega\omega}_x(\langle p, q\rangle) = f_M(\langle p, q\rangle) = f(y)$, so $F \subseteq range(\eta^{\omega\omega})$. $\square$

Lemma 6.6. *$utm(\eta^{ab})$ and $s^m_n(\eta^{ab})$ hold.*

Proof. $utm(\eta^{ab})$: By the $utm(\xi^{\omega b})$ there exists a computable partial function $v : \Sigma^* \times \Sigma^\omega \to \Sigma^b$ such that $v(x, y) = \xi^{\omega b}_x(y)$. Let $u : \Sigma^\omega \times \Sigma^a \to \Sigma^b$ be defined by $u(\langle x, p\rangle), y) := v(x, \langle p, y\rangle)$. So $\eta^{ab}_{\langle x,p\rangle}(y) = \xi^{\omega b}_x \langle p, y\rangle = v(x, \langle p, y\rangle) = u(\langle x, p\rangle, y)$.

$s^m_n(\eta^{ab})$: Let $v : \Sigma^\omega \times \Sigma^a \to \Sigma^b$ be computable. Then there is some $x \in \Sigma^*$ such that $\xi^{\omega b}_x \langle p, y\rangle = g(p, y)$. Let $s : \Sigma^\omega \to \Sigma^\omega$ be such that $s(p) := \langle x, p\rangle$ which is computable. So $g(x, y) = \xi^{\omega b}_x \langle p, y\rangle = \eta^{ab}_{\langle x,p\rangle}(y) = \eta^{ab}_{s(p)}(y)$. $\square$

N.B. a similar argument holds for the equivalence lemma as in 6.3.

6.3 The representation of continuous functions

Finally η^{ab} can be used to represent abstract functions. Without any further work beyond the definition below a representation $[\vartheta \to \rho^2]$ of the those cyclic functions continuous on the standard topologies of $\mathbb{T}$ and $\mathbb{C}$ is obtained. It is known that this is the case since $\eta^{\omega\omega}$ represents the continuous string functions and by admissibility of ϑ and ρ^2 with respect to the standard topologies and also in conjunction with Theorem 4.3 it is known that the resulting abstract functions are also continuous with respect to the standard topologies. Conveniently the computable functions are the computable elements of the representation.

As a side effect many more representations of cyclic functions are created each resulting in a different set of functions being represented derived from the distinct representations of $\mathbb{T}$ and $\mathbb{C}$.

Definition 6.7. *Let δ_i be a naming system of M_i for $i \in (n + 1)$ such that $\delta_0 : \Sigma^b \to M_0$ and $\langle s_i\rangle^n_{i=1} \in \Sigma^a$ where $a, b \in \{*, \omega\}$.*

Let $[(\delta_i)^n_{i=1} \to \delta_0]$ be the representation of $((\delta_i)^n_{i=1}, \delta_0)$-continuous functions in $M_0^{\prod^n_{i=1} M_i}$ defined by

$$[(\delta_i)^n_{i=1} \to \delta_0](p) = f \Leftrightarrow f((\delta_i(s_i))^n_{i=1}) = \delta_0 \circ \eta^{ab}_p \langle s_i\rangle^n_{i=1}$$

where $\forall i, s_i \in dom(\delta_i)$ and $dom(f) = \prod_{i=1}^{n} M_i$.

The superscript on η will be omitted in general from now on since the type can be deduced easily from the context and would clutter results. Here is a useful result on the continuous function representation.

Lemma 6.8. *$(g, f) \mapsto g \circ f$ is $([\epsilon \to \delta], [\gamma \to \epsilon], [\gamma \to \delta])$-computable.*

Proof. Let $[\epsilon \to \delta](q) = g$, $[\gamma \to \epsilon](p) = f$ and let $r \in dom(\gamma)$. So, $g \circ f \circ \gamma(r) = g \circ \epsilon \circ \eta_q(r)$, by definition of $[\gamma \to \epsilon]$, $= \delta \circ \eta_q \circ \eta_p(r)$, by definition of $[\epsilon \to \delta]$. By $utm(\eta)$ and $s_n^m(\eta)$ there exist computable functions a, b, c, d, e such that $\eta_q \circ \eta_p(r) = \eta_q \circ a(p, r) = b(q, a(p, r)) = c(d(q, p), r) = \eta_{e(q,p)}(r)$. $\square$

6.4 The Curry theorem

This theorem will be the most useful result from this point onwards and is a generalisation of [1] 3.3.15 type conversion. It is essentially a neat way of reasoning about the representation of continuous functions without having to worry about the representation itself but instead concerning the computability of the function represented.

Theorem 6.9. *Let δ_i be a representation of M_i for $i \in n+1$ and let $Y := \prod_{i=2}^{n} M_i$. For any total function $f : Y \times M_1 \to M_0$ there exists a transform T such that $T(f) : Y \to M_1^{M_0}$ where*

$$T(f)\left((y_i)_{i=2}^{n}\right)(x) := f\left((y_i)_{i=2}^{n}, x\right)$$

and also

$$f \text{ is } \left((\delta_i)_{i=2}^{n}, \delta_1, \delta_0\right)\text{-computable} \Leftrightarrow T(f) \text{ is } \left((\delta_i)_{i=2}^{n}, [\delta_1 \to \delta_0]\right)\text{-computable.}$$

Proof. Let $(s_i)_{i=1}^{n}$ be such that $s_i \in dom(\delta_i)$. Let $f : Y \times M_1 \to M_0$ and let $p \in \Sigma^\omega$ such that $[(\delta_i)_{i=1}^{n} \to \delta_0](p) = f$. Similarly, let $g : Y \to M_0^{M_1}$ and let $q \in \Sigma^\omega$ such that $[(\delta_i)_{i=2}^{n} \to [\delta_1 \to \delta_0]](q) = g$.

$$\begin{aligned}
&T \circ f\left((\delta_i(s_i))_{i=2}^{n}, \delta_2(s_2)\right)(\delta_1(s_1)) \\
=&T \circ [(\delta_i)_{i=1}^{n} \to \delta_0](p)\,(\delta_i(s_i))_{i=2}^{n}\,(\delta_1(s_1)), \text{by definition of } f, \\
=&\left[\left((\delta_i)_{i=2}^{n}, \delta_1\right) \to \delta_0\right](p)\left((\delta_i(s_i))_{i=1}^{n}\right), \text{by the definition of } T, \\
=&\delta_0 \circ \eta_p \langle s_i \rangle_{i=1}^{n}, \text{by definition of the representation of continuous functions.}
\end{aligned}$$

Also,

$$\begin{aligned}
&g\left(\left(\delta_i\left(s_i\right)\right)_{i=2}^n\right)\left(\delta_1\left(s_1\right)\right)\\
&=\left[\left(\delta_i\right)_{i=2}^n \to \left[\delta_1 \to \delta_0\right]\right](q)\left(\delta_i\left(s_i\right)\right)_{i=2}^n\left(\delta_1\left(s_1\right)\right), \text{by definition of } g,\\
&=\left(\left[\delta_1 \to \delta_0\right]\eta_q\left\langle s_i\right\rangle_{i=2}^n\right)\left(\delta_1\left(s_1\right)\right), \text{by definition of } \to,\\
&=\left(\delta_0 \circ \eta_{\eta_q\left\langle s_i\right\rangle_{i=2}^n}\right)\left(s_1\right), \text{again by definition of } \to.
\end{aligned}$$

Now, by $utm(\eta)$ there exists a computable function A such that, $\eta_p\left\langle s_i\right\rangle_{i=1}^n = A\left(\left\langle p, \left\langle s_i\right\rangle_{i=2}^n\right\rangle, s_1\right)$ and, by $s_n^m(\eta)$, there exist computable functions B, C such that $A\left(\left\langle p, \left\langle s_i\right\rangle_{i=2}^n\right\rangle, s_1\right) = \eta_{B\left\langle p, \left\langle s_i\right\rangle_{i=2}^n\right\rangle}\left(s_1\right) = \eta_{\eta_{C(p)}\left\langle s_i\right\rangle_{i=2}^n}\left(s_1\right)$. So $C(p) = q$, hence all such f computably translates to a g as above where both functions take on the same values for all possible sequences $\left(s_i\right)_{i=0}^{n-1}$ as above.

Conversely, by $utm(\eta)$ there exist computable functions D, E such that $\eta_{\eta_q\left\langle s_i\right\rangle_{i=1}^n}(s_1) = \eta_{D(q, \left\langle s_i\right\rangle_{i=2}^n)}(s_1) = E(D(q, \left\langle s_i\right\rangle_{i=2}^n), s_1)$. Since composition is closed under computability, there exists a computable function F such that $E(D(q, \left\langle s_i\right\rangle_{i=2}^n), s_1) = F(q, \left\langle s_i\right\rangle_{i=1}^n)$. Finally, by $s_n^m(\eta)$, there exists a computable function G such that $F(q, \left\langle s_i\right\rangle_{i=1}^n) = \eta_{G(q)}\left\langle s_i\right\rangle_{i=1}^n$. So $G(q) = p$ forms the converse translation function between g and f such that $g = T(f)$. $\square$

Corollary 6.10. *Let δ be a representation of some subset, N, of $M_1^{M_2}$, γ_i be a representation of M_i and let apply$(f, x) = f(x)$. apply is $(\delta, \gamma_1, \gamma_2)$-computable $\Leftrightarrow \delta \leq [\gamma_1 \to \gamma_2]$.*

This immediate corollary of the Curry theorem says that the standard representation of continuous functions is the weakest representation, i.e. gives rise to the largest number of computable functions, such that *apply* is computable. This observation will be useful in the next section for understanding that, for most of the representation defined there, *apply* is not computable which may appear to render the functions represented useless in a computable world. This is not entirely true as will be seen.

7 Representations of integrable functions

7.1 Complex sequence spaces

Complex valued sequence spaces are used in the process of defining more representations of cyclic functions. A sequence $(z_k)_{k\in\mathbb{Z}}$ of complex numbers can be regarded as a function from $\mathbb{Z}$ to $\mathbb{C}$ so the representation, $[\nu_{\mathbb{Z}} \to \rho^2]$, which exist due to the work done in the previous section, appears to be a suitable choice. The suitability of $[\nu_{\mathbb{Z}} \to \rho^2]$ is backed up by the first result below which shows that $[\nu_{\mathbb{Z}} \to \rho^2]$ is admissible. The results here make good use of results from the previous section which makes it worthwhile having generalised the definition of η to include finite inputs and infinite outputs.

Lemma 7.1. *If representation γ of X is admissible with respect to topology τ on X then $[\nu_{\mathbb{Z}} \to \gamma]$ is admissible with respect to the product topology generated by τ on $X^{\mathbb{Z}}$.*

Proof. Since γ is admissible with respect to τ then there exists an effective topological space $\mathbf{S} = (M, \sigma, \nu)$. Define an new effective topological space $\mathbf{S}^{\mathbb{Z}} = \left(M^{\mathbb{Z}}, \sigma^{\mathbb{Z}}, \nu^{\mathbb{Z}}\right)$ where $\sigma^{\mathbb{Z}} := \left\{J(U_k)_{|k|<N} : U_k \in \sigma \wedge N \in \mathbb{N}\right\}$ such that $J(U_k)_{|k|<N} := \left\{(z_k)_{k\in\mathbb{Z}} \in M^{\mathbb{Z}} : z_k \in U_k \wedge |k| < N\right\}$. Also let $\nu^{\mathbb{Z}} \langle u_k \rangle_{k=0}^{2N} = J(\nu(u_{t(k)}))_{|k|<N}$ where $u_k \in dom(\nu)$ and

$$t(k) = \begin{cases} 2k & \text{if } k \geq 0 \\ -2k-1 & \text{if } k < 0. \end{cases}$$

.

Firstly $A : \left((z_k)_{k\in\mathbb{Z}}, n\right) \mapsto z_n$ is $(\delta_{\mathbf{S}^{\mathbb{Z}}}, \nu_{\mathbb{Z}}, \gamma)$-computable since there exists a type-2 machine M which takes inputs $p \in dom(\delta_{\mathbf{S}^{\mathbb{Z}}})$ and $n \in range(\nu_{\mathbb{Z}})$ which searches for $\iota(u) \lhd p$. If $u = \langle w_i \rangle_{i=0}^{N}$ such that $N > 2n$ then write $\iota(w_{t(i)})$ to the output tape. So, by the Curry theorem, $(TA)\left((z_k)_{k\in\mathbb{Z}}\right)(n) = A\left((z_k)_{k\in\mathbb{Z}}, n\right) = z_n$ is $(\delta_{\mathbf{S}^{\mathbb{Z}}}, [\nu_{\mathbb{Z}} \to \gamma])$-computable hence $\delta_{\mathbf{S}^{\mathbb{Z}}} \leq [\nu_{\mathbb{Z}} \to \gamma]$.

Conversely, define a type-2 machine, M', which acts upon inputs $p \in dom\left([\nu_{\mathbb{Z}} \to \delta_{\mathbf{S}}]\right)$. $[\nu_{\mathbb{Z}} \to \delta_{\mathbf{S}}](p) = f \Leftrightarrow f(\nu_{\mathbb{Z}}(s)) = \delta_{\mathbf{S}} \circ \eta_p^{*\omega}(s)$ hence the machine can work in stages where at stage n it takes each $\nu_{\mathbb{Z}}^{-1}(k)$ where $|k| < n$ and evaluates $q := \eta_p^{*\omega}(\nu_{\mathbb{Z}}^{-1}(k))$ until the first n $\iota(w_{k,j}) \lhd q$ have been produced. Finally it extends the output tape with every string of the form $\iota\left(\left\langle w_{t^{-1}(k),j_k}\right\rangle_{k=0}^{2m}\right)$ where $m \leq n$ and $(\forall k) 0 < j_k \leq n$. Hence $f_{M'}$ translates $[\nu_{\mathbb{Z}} \to \delta_{\mathbf{S}}]$ to $\delta_{\mathbf{S}^{\mathbb{Z}}}$. Hence, by admissability of γ, $[\nu_{\mathbb{Z}} \to \gamma] \leq \delta_{\mathbf{S}^{\mathbb{Z}}}$. □

Lemma 7.2.

$$(z_i)_{i=0}^{n} \mapsto z := \lim_{i\to\infty} z_i$$

where $\|z_i - z\|_\infty \leq 2^{-i}$ is $\left([\nu_{\mathbb{N}} \to \rho^2], \rho^2\right)$-computable.

This is a simple consequence of the result for convergence of real numbers [1] theorem 4.3.7 since real and imaginary parts can be considered separately.

Lemma 7.3.

$$S : \left((z_i)_{i\in\mathbb{Z}}, e\right) \mapsto \sum_{i\in\mathbb{Z}} z_i$$

$$\left((z_i)_{i\in\mathbb{Z}}, e\right) \in dom(S) \Leftrightarrow (\forall m > n \geq e(N)) \left|\sum_{n+1<|i|<m} z_i\right| < 2^{-N}$$

is $([\nu_{\mathbb{Z}} \to \rho^2], [\nu_{\mathbb{N}} \to \nu_{\mathbb{N}}], \rho^2)$*-computable.*

Proof. It is clear that *apply* $\left((z_k)_{z\in\mathbb{Z}}, n\right) \mapsto z_n$ is $([\nu_{\mathbb{Z}} \to \rho^2], \nu_{\mathbb{Z}})$-computable by Corollary 6.10, similarly for $e(n)$, and multiplication is (ρ^2, ρ^2, ρ^2)-computable so primitive recursively define $h\left((z_k)_{k\in\mathbb{Z}}, 0\right) = z_0$ and $h\left((z_k)_{k\in\mathbb{Z}}, n+1\right) := h\left((z_k)_{k\in\mathbb{Z}}, n\right) + z_{n+1} + z_{-n-1}$ which preserves computability by [1] Theorem 3.1.7 hence is $([\nu_{\mathbb{Z}} \to \rho^2], \nu_{\mathbb{N}}, \rho^2)$-computable. So, by the Curry theorem and computable composition, $(Tg)\left((z_k)_{k\in\mathbb{Z}}\right) = \left(\sum_{|k|\leq e(n)} z_k\right)_{n\in\mathbb{N}}$ is $([\nu_{\mathbb{Z}} \to \rho^2], [\nu_{\mathbb{N}} \to \rho^2])$-computable, where $g\left((z_k)_{k\in\mathbb{Z}}, n\right) = h\left((z_k)_{k\in\mathbb{Z}}, e(n)\right)$. Finally, by an application of Lemma 7.2, which can be done since $(\forall m > n \geq e(N))2^{-N} > \left|\sum_{n+1<|i|<m} z_i\right|$ so $2^{-N} \geq \left|\sum_{e(N)+1<|i|} z_i\right| = \left|\sum_{i\in\mathbb{Z}} z_i - \sum_{|i|\leq e(N)}\right|$, the result holds. □

7.2 Representations of $\mathsf{L}(\mathbb{T})$

In this section a representation of the integrable functions such that integration itself is computable is constructed. A method similar to that in [4] used for constructing the set of Lebesgue integrable functions will be adopted as a model for this construction. Firstly a representation, $\lambda^{step}_{\gamma,\delta}$, of complex valued step functions on $\mathbb{T}$ is formulated. This is then used to produce a representation, $\lambda^{inc}_{\gamma,\delta}$, of integrable functions which can be represented by a sequence of step functions which are increasing in their real and imaginary parts and whose sequence of integrals a within a controlled distance of the final integral - $L^{inc}(\mathbb{T})$. Finally the representation, $\lambda^{1}_{\gamma,\delta}$, is created from $\lambda^{inc}_{\gamma,\delta}$ by considering the difference of $L^{inc}(\mathbb{T})$ functions. The computability of integration itself is verified in the process.

Definition 7.4. *Let γ be a representation of $\mathbb{T}$ and δ a representation of $\mathbb{C}$.*

$$\lambda^{step}_{\gamma,\delta}\left(\iota\left(\nu_{\mathbb{N}}^{-1}(k)\right)\langle u_i\rangle_{i=1}^{k}\right) = f \Leftrightarrow (\forall i)\, u_i = \langle l_i, h_i, r_i\rangle$$

where $l_i, r_i \in dom(\gamma)$, $h_i \in dom(\delta)$ and $\gamma(l_i) \neq \gamma(r_i)$ and also

$$f(\dot{x}) = \sum_{i=1}^{k} s(h_i)$$

where $s(h_i) = \begin{cases} \delta(h_i) & \textit{if } \dot{x} \textit{ is strictly between } l_i \textit{ and } r_i \textit{ anti-clockwise} \\ 0 & \textit{otherwise.} \end{cases}$

Theorem 7.5. *$I : f \mapsto \int f$ is $(\lambda^{step}_{\vartheta,\rho^2}, \rho^2)$-computable.*

Proof. For all i, each projection from $f = \lambda^{step}_{\vartheta,\rho^2}(\iota(\nu_{\mathbb{N}}^{-1}(k))\,\langle u_i\rangle_{i=1}^{k})$, where $u_i = \langle l_i, h_i, r_i\rangle$, to each of $\vartheta(l_i)$, $\vartheta(r_i)$, $\rho^2(h_i)$, is $(\lambda^{step}_{\vartheta,\rho^2}, \nu_{\mathbb{N}}, \vartheta)$-computable, $(\lambda^{step}_{\vartheta,\rho^2}, \nu_{\mathbb{N}}, \vartheta)$-computable, $(\lambda^{step}_{\vartheta,\rho^2}, \nu_{\mathbb{N}}, rho^2)$-computable respectively. Since $(x_i := \vartheta(l_i)) \neq (y_i := \vartheta(r_i))$, $\Lambda(x_i, y_i)$ is $(\vartheta, \vartheta, \rho)$-computable. Also multiplication is (ρ^2, ρ^2, ρ^2)-computable and $\rho \leq \rho^2$ hence, since composition preserves computability, $(x_i, z_i, y_i) \mapsto z_i\Lambda(x_i, y_i)$, where $z_i = \rho^2(h_i)$, is $(\vartheta, \rho^2, \vartheta, \rho^2)$-computable. Finally, since $(a, b) \mapsto a + b$ is (ρ^2, ρ^2, ρ^2)-computable, by k-fold application and preservation of computability through composition the proposition holds. $\square$

Definition 7.6. *Let $\lambda^{inc}_{\gamma,\delta}(\langle p_i\rangle_{i=1}^{\infty}) = f$ where $\forall i, p_i \in dom(\lambda^{step}_{\gamma,\delta})$ if and only if the following are satisfied:*

$$\begin{aligned} f &= \lim_{i\to\infty} \lambda^{step}_{\gamma,\delta}(p_i) \\ \forall i, &\left|\int \left(f - \lambda^{step}_{\gamma,\delta}(p_i)\right)\right| < 2^{-i} \\ \forall i, &\Re \circ \lambda^{step}_{\gamma,\delta}(p_i) \leq \Re \circ \lambda^{step}_{\gamma,\delta}(p_{i+1}) \\ \forall i, &\Im \circ \lambda^{step}_{\gamma,\delta}(p_i) \leq \Im \circ \lambda^{step}_{\gamma,\delta}(p_{i+1}). \end{aligned}$$

Theorem 7.7. *$I : f \mapsto \int f$ is $(\lambda^{inc}_{\vartheta,\rho^2}, \rho^2)$-computable.*

Proof. The projection $(f, i) \mapsto \phi_i$ where $f = \lim_{i\to\infty} \phi_i$ and $\left|\int(\phi_i - f)\right| < 2^{-i}$ is $(\lambda^{inc}_{\vartheta,\rho^2}, \nu_{\mathbb{N}}, \lambda^{step}_{\vartheta,\rho^2})$-computable. In addition, by Theorem 7.5 and preservation of computability under composition, $(f, i) \mapsto \int \phi_i$ is $(\lambda^{inc}_{\vartheta,\rho^2}, \nu_{\mathbb{N}}, \rho^2)$-computable. By the Curry theorem, $f \mapsto \left(\int \phi_i\right)_{i=1}^{\infty}$ is $(\lambda^{inc}_{\vartheta,\rho^2}, [\nu_{\mathbb{N}} \to \rho^2])$-computable. Since $\left|\int f - \int \phi_i\right| = \left|\int(f - \phi_i)\right| < 2^{-i}$, by Theorem 7.2 and computable composition, $f \mapsto \lim_{i\to\infty}\left(\int \phi_i\right) = \int f$ is $(\lambda^{inc}_{\vartheta,\rho^2}, \rho^2)$-computable. $\square$

Definition 7.8. *Let $\lambda^{1}_{\gamma,\delta}(\langle p, q\rangle) = f \Leftrightarrow f = \lambda^{inc}_{\gamma,\delta}(p) - \lambda^{inc}_{\gamma,\delta}(q)$.*

Theorem 7.9. *$I : f \mapsto \int f$ is $(\lambda^{1}_{\vartheta,\rho^2}, \rho^2)$-computable.*

Proof. The two projections $(\lambda^{1}_{\vartheta,\rho^2}(\langle p, q\rangle)) \mapsto \lambda^{inc}_{\vartheta,\rho^2}(p), \lambda^{inc}_{\vartheta,\rho^2}(q)$ are $(\lambda^{1}_{\vartheta,\rho^2}, \lambda^{inc}_{\vartheta,\rho^2})$-computable. Since subtraction is (ρ^2, ρ^2, ρ^2)-computable, by Theorem 7.7 and computable composition, the proposition holds. $\square$

7.3 Relationship between $\lambda^1_{\gamma,\delta}$ and $[\gamma \to \delta]$

Several new representations of some subset of the cyclic functions have been obtained with thus far one computable function defined i.e. integration. It is obvious that $\lambda^{step}_{\vartheta,\rho^2} \le \lambda^{inc}_{\vartheta,\rho^2} \le \lambda^{1}_{\vartheta,\rho^2}$. The following results show that the continuous function representations tie in through computable reduction to $\lambda^{inc}_{\vartheta,\rho^2}$. This shows that integration is also computable on $[\vartheta \to \rho^2]$ by converting the function to the $\lambda^{inc}_{\vartheta,\rho^2}$ representation first. However *apply* is not computable in any of these new representations by Corollary 6.10.

One of the most interesting aspects of this paper is that compactness of $\mathbb{T}$ is used to ensure that the reduction is successful hence it is unlikely that there is an analogous reduction had this paper been concerned with functions with domain $\mathbb{R}$ for example.

Lemma 7.10. *An arbitrary choice function of multi-valued function* $(f,n) \mapsto \psi_n$ *such that* $\|\psi_n - f\|_\infty < 2^{-n}$ *is* $\left([\vartheta \to \rho^2], \nu_{\mathbb{N}}, \lambda^{step}_{\vartheta,\rho^2}\right)$*-computable.*

Proof. Since $\rho^2 \equiv \rho^2_C$ and $\vartheta \equiv \vartheta_C$, by Theorem 4.22, $[\vartheta \to \rho^2] \equiv [\vartheta_C \to \rho^2_C]$. Let $[\vartheta_C \to \rho^2_C](p) = f$. Let N be the type-2 machine behind $utm(\eta^{\omega\omega})$ in Lemma 6.6.

Let a type-2 machine, M, systematically work through each $v \in L$ where $L := \left\{\oplus^k_{i=1}\iota(u_i) : \oplus^k_{i=1}\iota(u_i) \sqsubseteq q \in dom(\vartheta_C)\right\}$ which is recursive since the conditions $u_i \in dom(\nu_{\mathbb{Q}})$ and $\forall m, m' > i, \|\dot{\nu_{\mathbb{Q}}}(u_m) - \dot{\nu_{\mathbb{Q}}}(u_{m'})\| < 2^{-i}$ can be checked for finite initial segments as a consequence of Lemma 4.23.

At each stage, M simulates N on input (p, v) until N requests an input beyond v upon which point it considers the output so far generated, say w, for which it is known that there exists $q_1, q_2 \in dom(\rho_C)$ such that $w \sqsubseteq \langle q_1, q_2\rangle$ of which say $w_1 \sqsubseteq q_1$ and $w_2 \sqsubseteq q_2$ can be determined and from that the last $\iota(h_1) \lhd w_1$ and $\iota(h_2) \lhd w_2$ can be found as well as checking that there are at least n distinct $\iota(h^*)$'s before each of $\iota(h_1)$ and $\iota(h_2)$. If that final check fails then move onto the next $v' \in L$ and start again. Otherwise let $z = \nu_{\mathbb{Q}}(h_1) + i\nu_{\mathbb{Q}}(h_2)$ and consider $v = \oplus^k_{i=1}(u_i)$ and $\mathbf{B} := \bigcap^k_{i=1} B\left(\dot{\nu_{\mathbb{Q}}}(u_i), 2^{-i}\right)$ and finds the end points $l, r \in \mathbb{Q}$. The end point r can be computed by initializing it to $\nu_Q(u_1) - 2^{-1}$ and considering at stage $i = 1$ to k whether $\dot{\nu_{\mathbb{Q}}}(u_i) \dot{-} 2^{-i}$ is anti-clockwise between $\dot{r}$ and $\dot{\nu_{\mathbb{Q}}}(u_i)$, which is decidable for rationals by Lemma 4.23, if so updating r to $\nu_{\mathbb{Q}}(u_i)$. Similarly for l.

Now, $f[\mathbf{B}] = f \circ \vartheta_C(v\Sigma^\omega) \subseteq \rho^2_C \circ \eta_p[v\Sigma^\omega] \subseteq \rho^2_C[w\Sigma^\omega] \subseteq B(z, 2^{-n})$ hence a single step within the required proximity to f along its entire length, i.e. the step of height z starting at $\dot{l}$ and going anti-clockwise round $\mathbb{T}$ to $\dot{r}$, has been found. Furthermore this information can be recorded on the "storage tape" in finite space since it is notated by a finite set of elements of $dom(\nu_{\mathbb{Q}})$.

Now consider all such triples recorded so far on the storage tape. Take for instance (l', h', r'): If both l', r' are anti-clockwise between l, r then delete (l', h', r') from the storage tape. If only l' is then include $(l', -h', r)$ on the storage tape and if only r' is then include $(l, -h', r')$. Hence any overlaps with previously determined steps will not distort the step calculated at the current stage of the computation.

Finally, each anti-clockwise interval (l_i, r_i) recorded is open and $\mathbb{T} \subseteq \bigcup_{i=1}^{\infty}(l_i, r_i)$ but $\mathbb{T}$ is compact hence there exists $I \in \mathbb{N}$ such that $\mathbb{T} \subseteq \bigcup_{i=1}^{I}(l_i, r_i)$. Hence, if at the end of each stage M checks the predicate $P \cong \mathbb{T} \subseteq \bigcup\{(l, r) : (l, h, r) \text{ is on the storage tape}\}$ then it will eventually hold upon which point the number of triples on the storage tape are counted, say k, and the tuples, say $(l_i, h_i, r_i)_{i=1}^{k}$ are wrapped up in the form $p^* := \nu_{\mathbb{N}}^{-1}(k)\left\langle \vartheta^{-1}(l_i), (\rho^2)^{-1}(h_i), \vartheta^{-1}(r_i)\right\rangle_{i=1}^{k} \in \lambda_{\vartheta,\rho^2}^{step}$.

Note that P is decidable since any $\dot{l_0}$ can be chosen. The machine can then work in stages where at stage $(j+1)$ each potential $\dot{l}_{j+1}$ is considered until one is found such that such that l_{j+1} is anti-clockwise between $\dot{l_j}$ and $\dot{r_j}$ and also $\dot{r}_{j+1}$ is anti-clockwise between $\dot{r_j}$ and $\dot{l}_{j+1}$. M then continues to the next stage unless either no such l_{j+1} is found, in which case it rejects the predicate, or $\dot{r}_{j+1}$ is anti-clockwise between $\dot{l_0}$ and $\dot{l}_{j+1}$ in which case it accepts P.

□

Theorem 7.11. $[\vartheta \to \rho^2] \leq \lambda_{\vartheta,\rho^2}^{inc}$

Proof. Let $[\vartheta \to \rho^2](p) = f$. By Lemma 6.8 $f - (1+i)2^{-i-1}$ is $[\vartheta \to \rho^2]$-computable, since $-(1+i)2^{-i-1}$ is ρ^2-computable and addition is (ρ^2, ρ^2, ρ^2)-computable. By Lemma 7.10 and composition, the mapping $(f, i) \mapsto \psi_i$ is $\left([\vartheta \to \rho^2], \nu_{\mathbb{N}}, \lambda_{\vartheta,\rho^2}^{step}\right)$-computable such that $\psi_i \in dom(\lambda_{\vartheta,\rho^2}^{step})$ and $\|\psi_i - (f - (1+i)2^{-i-1})\|_\infty < 2^{-(i+3)} \Rightarrow \|\psi_i - f\|_\infty < 2^{-i}$ which ensures that the step functions are increasing with i and also converge to f at infinity.

Finally f is both continuous and has a compact domain hence is integrable so $\int f$ exists. So $|\int f - \int \psi_{i+3}| = |\int (f - \psi_{i+3})| \leq \int \|f - \psi_{i+3}\|_\infty \leq 2^{-i-3}2\pi \leq 2^{-i}$. Hence $q = \left\langle \left(\lambda_{\vartheta,\rho^2}^{step}\right)^{-1}(\psi_i)\right\rangle_{i=4}^{\infty} \in dom\left(\lambda_{\vartheta,\rho^2}^{inc}\right)$ can clearly be output by a type-2 machine and $\lambda_{\vartheta,\rho^2}^{inc}(q) = f$. □

Theorem 7.12. $\neg\left(\lambda_{\vartheta,\rho^2}^{inc} \leq_t [\vartheta \to \rho^2]\right)$

Proof. This is a consequence of appealing to topology again through Lemma 4.12 since for example $p\left(\chi_{(-1,1)}\right)$ is in $L^{inc}(\mathbb{T})$ but not in $C(\mathbb{T})$. □

8 Fourier representations

8.1 Absolutely convergent Fourier series

To wrap up this paper the focus on cyclic functions will be taken advantage of to construct a representation which is based on absolutely convergent Fourier series. Absolute convergence is probably the neatest condition which can be put on sequences of complex numbers to ensure the pointwise convergence of Fourier series hence will be encapsulated in the representation.

Definition 8.1. *Let the absolutely convergent Fourier series representation, α_γ, be such that $\alpha_\gamma\left(\langle q, \langle p_i\rangle_{i=0}^{\infty}\rangle\right) = f \Leftrightarrow f = \sum_{k\in\mathbb{Z}} c_k e_k$ and*

$$\forall m, n, m > n > r(N) \Rightarrow \sum_{n<|k|<m} |c_i| < 2^{-N}$$

where $r = [\nu_{\mathbb{N}} \to \nu_{\mathbb{N}}]\,(q)$ and $\forall i, c_i = \gamma(p_i)$.

Theorem 8.2. $\alpha_{\rho^2} \leq [\vartheta \to \rho^2]$

Proof. $(n, x) \mapsto e_n(x)$ is $(\nu_{\mathbb{Z}}, \vartheta, \rho^2)$-computable since composition preserves computability and by the following ingredients: multiplication is $(\nu_{\mathbb{Z}}, \vartheta, \vartheta)$-computable by Theorem 4.26 and $\theta \mapsto \exp(i\theta)$ is (ϑ, ρ^2)-computable as discussed in the aftermath of Lemma 5.5.

Where $f = \sum_{k\in\mathbb{Z}} c_n e_n$ where $r : \mathbb{N} \to \mathbb{N}$ such that $\forall m > n > r(N)$, $\sum_{n<|k|<m} |c_k| < 2^{-N}$ it is clear that the sequences of strings in its α_{ρ^2} representation can be unfolded such that projections, $(f, k) \mapsto c_k$, is $(\alpha_{\rho^2}, \nu_{\mathbb{Z}}, \rho^2)$-computable and $(f, k) \mapsto r$ is $(\alpha_{\rho^2}, \nu_{\mathbb{Z}}, [\nu_{\mathbb{N}} \to \nu_{\mathbb{N}}])$-computable.

Since, by [1] 4.3.9, multiplication is (ρ^2, ρ^2, ρ^2)-computable there exists a $G : (f, k) \mapsto c_k e_k(x)$ is $(\alpha_{\rho^2}, \nu_{\mathbb{Z}}, \rho^2)$-computable, since composition preserves computability. By the Curry theorem $T\,(G)\,(f, x)\,(n) = G\,(f, x, n)$ is $(f, [\nu_{\mathbb{Z}} \to \rho^2])$-computable. By Lemma 7.3, since whenever $n > m > r(N)$, $2^{-N} > \sum_{m<|k|<n} |c_k| = \sum_{m<|k|<n} |c_k e_k| > \left|\sum_{m<|k|<n} c_k e_k\right|$, and by computable composition, $H(f, x) := S\left(\left(T\,(G)\,(f, x)_{k\in\mathbb{Z}}\right), r\right) = \sum_{k\in\mathbb{Z}} c_k e_k(x)$ is $(\alpha_{\rho^2}, \vartheta, \rho^2)$-computable.

A further application of the Curry theorem completes the proof since $(U \circ H)(f)(x) = H(f, x)$ so $(U \circ H)(f) = f$. □

Here it is interesting to note that the specification for the representation of α_{ρ^2} cannot greatly be improved by removing part relating to the modulus function $\mathbb{N} \to \mathbb{N}$ which was not required in the analogous part of ρ_C, ϑ_C, etc. This is because by uniqueness of the function generated by a Fourier series (see [6] 2.4) the modulus function cannot be fixed without restricting the set of functions represented. Also the modulus cannot be determined from the sequence therefore omitted entirely as shown below.

Lemma 8.3. *There is no* $([\nu_{\mathbb{Z}} \to \rho^2], [\nu_{\mathbb{Z}} \to \rho^2])$*-computable choice function such that* $(c_k)_{k\in\mathbb{Z}} \mapsto r$ *where* $\forall m > n > r(N), \sum_{n<|k|<m} |c_k| < 2^{-N}$.

Proof. It is sufficient to show there is no such continuous function.

Suppose F is $([\nu_{\mathbb{Z}} \to \rho^2], [\nu_{\mathbb{Z}} \to \rho^2])$-continuous such that $F : (c_k)_{k\in\mathbb{Z}} \mapsto r$ where $(\forall m > n > r(N)) \sum_{n<|k|<m} |c_k| < 2^{-N}$.

It is clear that $G_c : \mathbb{C} \to \ell^1(\mathbb{C})$ such that $G_c(a) = ac$ where $c \in \ell^1(\mathbb{C})$ is continuous for all such c hence, by Theorem 4.3 and 7.1, G_c is $(\rho^2, [\nu_{\mathbb{Z}} \to \rho^2])$-continuous. Hence, since composition preserves continuity, $H : a \mapsto F(G_c(a))(0)$ is $(\rho^2, \nu_{\mathbb{N}})$-continuous.

By Theorem 4.3, standard topological concepts can be used. Let N_0 be such that $H(a_0) = N_0$ where $a_0 \in \mathbb{C}$ so, since $\mathbb{N}$ has the discrete topology to allow admissibility, $\{N_0\}$ and $\mathbb{Z} - \{N_0\}$ are open so $U_1 := H^{-1}(\{N_0\})$ and $U_2 := H^{-1}(\mathbb{N} - \{N_0\})$ are open in $\mathbb{C}$ and also cover $\mathbb{C}$ since H is total so $U_2 = \mathbb{C} - U_1$. But $\mathbb{C}$ is connected and $a_0 \in U_1$ so $U_1 = \mathbb{C}$. Hence $H(a) = N_0$.

Hence, by definition of H, $\forall a \in \mathbb{C}$ $(\forall m > n > N_0) \sum_{n<|k|<m} |ac_k| < 2^{-0} = 1$ which can only hold for finite sequences, since $|a|$ can be made arbitrarily large, yielding a contradiction. Hence F does not exist in general. $\square$

8.2 Fourier transforms

An appropriate final note for a paper on computable cyclic functions is to consider the computability of the Fourier transform. This result has significant note in terms of the themes of this paper in that again functions are being manipulated in a computable manner without considering or even being able to consider the values which the function takes. This demonstrates part of the computable world which has been opened up as a result of the work in this paper.

Lemma 8.4.
$(\psi, \phi) \mapsto \psi + \phi$ *and* $(\psi, \phi) \mapsto \psi \cdot \phi$ *are both* $\left(\lambda^{step}_{\vartheta,\rho^2}, \lambda^{step}_{\vartheta,\rho^2}, \lambda^{step}_{\vartheta,\rho^2}\right)$*-computable.*

Proof. Trivially, addition can be realised by a type-2 machine which outputs a step function which consists of the steps every step in ψ and every step in ϕ.

Multiplication can be realised by a type-2 machine which outputs a step function which consists of the steps produced by multiplying every step in ψ by every step in ϕ. Multiplication of steps can be performed by multiplying the complex numbers which the steps take and finding the anti-clockwise start and end points where the steps intersect on $\mathbb{T}$. $\square$

Theorem 8.5. *The mapping $f \mapsto \hat{f}$ is $\left(\lambda^1_{\vartheta,\rho^2}, [\nu_{\mathbb{N}} \to \rho^2]\right)$-computable.*

Proof. Since e_n is $[\vartheta \to \rho^2]$-computable as shown in the proof of theorem 8.2 and $[\vartheta \to \rho^2] \leq \lambda^{inc}_{\vartheta,\rho^2}$, e_n is $\lambda^{inc}_{\vartheta,\rho^2}$-computable.

Following from the definitions of the λ representations, consider a function $f \in L^1(\mathbb{T})$ such that $\lambda^1_{\vartheta,\rho^2}(\langle p, q\rangle) = f = g - h$ where $g = \lambda^{inc}_{\vartheta,\rho^2}(p)$ and $h = \lambda^{inc}_{\vartheta,\rho^2}(q)$. Also let $\lambda^{inc}_{\vartheta,\rho^2}(\langle s_i\rangle_{i=1}^{\infty}) = g$ where $\forall i, s_i \in dom(\lambda^{step}_{\vartheta,\rho^2})$ and $\psi_i = \lambda^{step}_{\vartheta,\rho^2}(s_i)$ and similarly let $\lambda^{inc}_{\vartheta,\rho^2}(\langle t_i\rangle_{i=1}^{\infty}) = h$ where $\forall i, t_i \in dom(\lambda^{step}_{\vartheta,\rho^2})$ and $\phi_i = \lambda^{step}_{\vartheta,\rho^2}(t_i)$.

Considering real and imaginary parts of f separately and symmetrically as follows with consideration to the increasing properties of the step functions and the proximity of the integrals of the step function to the final integral.

$$\begin{aligned}
\|\Re f - (\Re\psi_{i+1} - \Re\phi_{i+1})\|_1 &= \int |(\Re g - \Re h) - (\Re\psi_{i+1} - \Re\phi_{i+1})| \\
&= \int |(\Re h - \Re\psi_{i+1}) - (\Re g - \Re\phi_{i+1})| \\
&\leq \int |(\Re h - \Re\psi_{i+1})| - \int |(\Re g - \Re\phi_{i+1})| \\
&= \left(\int \Re g - \int \Re\psi_{i+1}\right) - \left(\int \Re h - \int \Re\phi_{i+1}\right) \\
&\leq 2\sqrt{2}^{-1} 2^{-i-1} \\
&= \sqrt{2}^{-1} 2^{-i}
\end{aligned}$$

Hence $\forall i, \|f - (\psi_{i+1} - \phi_{i+1})\|_1 \leq 2^{-i}$.

Since integration is $\left(\lambda^1_{\vartheta,\rho^2}, \rho^2\right)$-computable a type-2 machine can compute an $m \in \mathbb{N}$ s.t. $\|\psi_i - \phi_i\|_1 = \int |\psi_i - \phi_i| \leq 2^m$.

Again following from the $\lambda^{inc}_{\vartheta,\rho^2}$ representation let $\lambda^{inc}_{\vartheta,\rho^2}(\langle u_i\rangle_{i=1}^{\infty}) = e_n$ where $\forall i, u_i \in dom(\lambda^{step}_{\vartheta,\rho^2})$ and $\theta_i = \lambda^{step}_{\vartheta,\rho^2}(u_i)$. By the linearity of integrals, the triangle inequality and the estimation theorem (see [5]) and also the above observations the following can be deduced

$$\begin{aligned}
&\left|\int fe_n - \int (\psi_{i+2} - \phi_{i+2})\, \theta_{i+m+1}\right| \\
&= \left|\int (f - (\psi_{i+2} - \phi_{i+2}))\, e_n + \int (\psi_{i+2} - \phi_{i+2})\, (e_n - \theta_{i+m+1})\right| \\
&\leq \left|\int (f - (\psi_{i+2} - \phi_{i+2}))\, e_n\right| + \left|\int (\psi_{i+2} - \phi_{i+2})\, (e_n - \theta_{i+m+1})\right| \\
&\leq \|f - (\psi_{i+2} - \phi_{i+2})\| + \|e_n - \theta_{i+m+1}\|_\infty \|(\psi_{i+2} - \phi_{i+2})\|_1 \\
&\leq 2^{-i-1} + 2^{-i-m-1}2^m \\
&= 2^{-i}.
\end{aligned}$$

So, a type-2 machine, M, which when input a representation of an integer, n, and with the concrete string representing f under the representation $\lambda^1_{\vartheta,\rho^2}$ can find the bound m as above demonstrated above and at stage i can find the strings for the strings for the step functions $\psi_{i+3}, \phi_{i+3}, \theta_{i+m+2}$ by the definition of the λ representations. It can then calculate $(\psi_{i+3} - \phi_{i+3})\, \theta_{i+m+2}$ by lemma 8.4 and $\int (\psi_{i+3} - \phi_{i+3})\, \theta_{i+m+2}$ by theorem 7.5 from which can be extracted through, translation between Cauchy representations, a rational within 2^{-i-1} precision to the result. Hence can be output eventually in an initial segment of an element of $dom(\rho^2_C)$ hence of $dom(\rho^2)$.

So f_M realises a function $(f, n) \mapsto \int fe_n$ which is $\left(\lambda^1_{\vartheta,\rho^2}, \nu_{\mathbb{Z}}, \rho^2\right)$-computable hence by one final application of the Curry theorem 6.9 the Fourier transform is $\left(\lambda^1_{\vartheta,\rho^2}, [\nu_{\mathbb{Z}} \to \rho^2]\right)$-computable. $\square$

9 Conclusion

Over the course of this paper several representations of cyclic functions have been defined and the tools for building many more have been created. However it would be nice to be able to draw them together in some reasonable manner involving the comparison of the representations through computable reduction forming a picture of how they relate in a partial order. It has become clear that this partial order is on a transfinite set of distinct classes of representations and that it is not linear and moreover it is a lattice by Lemmas 4.15 and 4.21. However, given the time available for the completion of this paper, not all of them could be considered.

Klaus Weihrauch had already given part of the partial order on the representation of the reals from which a hierarchy could be deduced for the complex numbers and use similar techniques to obtain a hierarchy on $\mathbb{T}$. On

$\mathbb{T}$ it was shown that ϑ had a number of equivalent inter translatable representations such as ϑ_C as shown in Theorem 4.19. Also the other representations considered sat above ϑ in the partial order hence had more computable objects but lacked the property of admissibility with respect to the standard topology on $\mathbb{T}$, a property which was seen to retain topological concepts in the representation.

So a point to begin considering the cyclic function representations needed to be chosen. It was known that only a subset size 2^ω could be considered and also that the space of continuous functions was both equinumerous to Σ^ω and occurred frequently in analysis. It was also observed that computability and continuity were closely related concepts. Given the nice relationship between topological continuity and continuity in the Cantor space which arose when a representation had the admissible property (see 4.3) and also given that the standard representation of continuous functions covered the space of continuous cantor functions, the representation $[\vartheta \to \rho^2]$ was a natural choice since the aforementioned observations meant that the space of continuous cyclic functions were being represented. However effectively an infinite class of representations were being considered already since the representation $\xi^{\omega\omega}$ was defined implicitly, rather than classical constructive methods, not to mention those representations obtained be replacing ϑ and ρ^2 with equivalent representations such as ϑ_C or $\vartheta_< \vee \vartheta_>$.

A small selection of distinct classes of representations which, through the course of the investigation, were found to relate to $[\vartheta \to \rho^2]$ by computable reduction, and hence which reside in the uncountable lattice of representations of cyclic functions, are shown in the diagram below. If there is a path upward to a representation, then more cyclic functions are computable in the higher representation since computable elements of the lower representation can be computably translated to the higher one therefore retaining the computability of the element but the converse does not hold. Also, if some operation is computable for one representation then it will remain computable for anything below it in the representation hierarchy. In corollary 6.10 the upper bounds for *apply* were established and in the closing sections the upper bounds for integration were considered.

$$\begin{tikzcd}
\lambda^{1}_{\vartheta,\rho^2} & {[\vartheta \to \rho^2_{\top}]} & \\
\lambda^{inc}_{\vartheta,\rho^2} \arrow[u] & {[\vartheta \to [\rho, \rho_{\top}]]} \arrow[u] & {[\vartheta \to [\rho_{\top}, \rho]]} \arrow[ul] \\
\lambda^{step}_{\vartheta,\rho^2} \arrow[u] & {[\vartheta \to \rho^2]} \arrow[ul] \arrow[u] \arrow[ur] & \\
\alpha_{\rho^2} \arrow[ur] & {[\vartheta_{<} \to \rho^2]} \arrow[u] & {[\vartheta_{>} \to \rho^2]} \arrow[ul] \\
 & {[\vartheta_{Cn} \to \rho^2]} \arrow[u] \arrow[ur] & {[\vartheta \to \rho^2_{\perp}]} \arrow[uul] \\
 & {[\vartheta_{Cn} \to \rho^2_{\perp}]} \arrow[u] \arrow[ur] &
\end{tikzcd}$$

Note that this is nowhere near the end of representations which have been either explicitly or implicitly constructed in this paper; for example $\alpha_{\rho^2} \wedge [\vartheta_{Cn} \to \rho^2_{\perp}]$ which is a lower bound for the representations shown in the diagram above or $\left[\vartheta \to \nu^2_{\mathbb{Q}}\right]$ which is the set of continuous functions taking complex rational values (which is the set of constant functions taking rational complex value by [1] Corollary 3.2.13) and satisfies $\left[\vartheta \to \nu^2_{\mathbb{Q}}\right] < [\vartheta \to \rho^2]$ or even the absolute bottom element the empty representation where nothing is represented let alone computable. However this hierarchy can be extended arbitrarily further upwards for instance consider $\upsilon := \lambda^1_{\vartheta,\rho^2} \vee [\vartheta \to \rho^2_{\top}]$ which is a representation which defines more functions than any of the representations shown above so, from Lemma 4.21 the largest set of function so far in this paper have become computable. Why not consider representation $\upsilon^+ = \upsilon \vee \epsilon$ where ϵ has the domain containing only the string 0^ω which maps to f where f is a unique cyclic function which is not in the range of υ so $\upsilon < \upsilon^+$. The set of computable functions could be extended in this contrived manner indefinitely but the point is that useful functions such as integration cease to be computable so there is little to be gained by considering them unlike with the representations focused on by this paper which have sought to maximise the set of computable functions while retaining utility in computable analysis.

Finally, the choice of model has paid off as it challenges ideas in other computability theories. For example computable implies continuous is not necessarily preserved when considering the abstract meaning of a computation as confirmed by the λ representation or it has been possible to compare say the $\vartheta, \vartheta_<$ and ϑ_{Cn} representations which would often be indistinguishable. More poignantly, in [7], the following claim is made:

> We show that a function f may be computable yet have a Fourier transform $\hat{f}$ which is not computable; and conversely $\hat{f}$ may be computable yet f not be computable.

This statement however conflicts with Theorem 8.5 since any $\lambda^1_{\vartheta,\rho^2}$-computable function can be transformed computably hence has a $[\nu_{\mathbb{Z}} \to \rho^2]$-computable Fourier transform. This apparent contradiction lies in the inability to change the representation in the axiomatic approach to computable analysis in [8] upon which [7] is based.

9.1 Where to next?

This paper has only touched upon the possible investigation paths opened up by considering Fourier analysis in the TTE model. Indeed in a sense the theory has not even been touched since all that has been done is to establish a solid foundation for the study of cyclic functions in a computable universe. In addition to the obvious route of incorporating more and more results in Fourier analysis into the theory several other question have arisen about the material covered. To conclude with some brief comments on some of these questions are noted.

- Can the admissibility of a quotient space representation which is obtained from two admissible representations be generalised to cover more than just $\mathbb{T}$? This would replace the chapter on $\mathbb{T}$ with a general theory and an acknowledgement that results on $\mathbb{R}$ and $\mathbb{Z}$ had already been established.

- Is λ^1 the weakest representation such that integration is computable or can we go weaker still? It would appear that having an extra two values for each step stating the value at it's end point instead of setting them to 0 every time would increase the number of computable functions but would only have cluttered the theory. It is likely that no weaker representation preserves the computability of integration.

- Vasko Brattka [2] observes that in modifying the representation $\rho_<$ such that it contains an integer bound at the beginning of the numbers allows a computable version of the uniform boundedness theorem which would normally be considered to be non-computable. This could then be adapted to cyclic functions which would allow more sequence space representations to be tied in which could be used to solve certain problems particularly those dependent on the uniform boundedness theorem and its corollaries.

- A non trivial representation of $\mathbb{T}$ and $\mathbb{C}$, say $\check{\vartheta}$ and $\check{\rho}^2$, be defined such that $\lambda^1_{\check{\vartheta},\check{\rho}^2} \equiv \left[\check{\vartheta} \to \check{\rho}^2\right]$? This would be interesting to be able to encode integrable functions in a continuous manner and use *apply*. This may be possible using representations with given primitive recursive convergence rates and allowing the use of recursive functions in evaluation since at discontinuous points the value can be drawn towards some predetermined point if convergence is not clear.

- The partial order structure is an endless source of investigation. What about $\alpha_{\rho_\perp}$ or $\lambda^1_{\vartheta_<,\rho^2_>}$? How do they tie in to other functions? Or perhaps how does $[\rho_\top, \vartheta_{Cn}]$ compare to $[\rho_\top, \rho_\top]$?

- What additional conditions are needed on a sequence of integrable functions to obtain a computable version of the monotone/dominated convergence theorem for $\lambda^1_{\vartheta,\rho^2}$?

- Can $\lambda^1_{\vartheta,\rho^2}$ be restricted to representing $\mathsf{L}^2(\mathbb{T})$ and obtain a computable reduction either way between itself and a representation of square summable sequence of complex numbers i.e. bringing the theory of Hilbert spaces into this computable world?

- While ϑ_{Cn} was shown to induce the trivial topology on $\mathbb{T}$ can the representations of $\mathbb{T}$ be balanced by a representation with a coarser induced topology. Intuitively it should come from the Cantor topology and would be such that only one string maps to an element of $\mathbb{T}$ or perhaps it would be such that no two distinct strings which share an initial segment can map to the same element. A weaker question would simply ask for any representation of $\mathbb{T}$, $\vartheta_\perp$ such that $\vartheta_\perp < \vartheta$.

- What are the complexities of the underlying algorithms and can they be improved upon? This has not been an issue since only the existence of an algorithm has been required. Of particular interest would be the algorithm in Lemma 7.10 since termination occurs after an arbitrary period of time due to the reliance on the compactness of $\mathbb{T}$ for termination.

- How can Weihrauch's model be formalised further and constructed in the theory of Hoare triple specification statements program refinement and data refinement under development in Oxford? Loop invariants can be used to ensure properties hold and would be a neat way of reasoning about epsilon style arguments in a computable environment. Can these theories deal with infinite inputs and outputs and infinite representations? The representations would appear to fit in nicely in the form of coupling invariants and data type invariants.

- What are the consequences of scrutinising the partial order of representations of cyclic function using domain theory? It has already been noted that the partial order is a lattice and has a bottom element but no top element. Is it for instance a complete partial order?

Finally, it would be of particular interest to follow through with the idea pointed out in response to Theorem 6.3 where generalised utm and s^m_n properties are used to non-deterministically obtain notations of computable string functions, a model which would appear to be extendable to deal with the mind bending idea of computations with uncountable strings as inputs and outputs. Perhaps a "representation" of all the cyclic functions can be obtained after all.

References

[1] Weihrauch, Klaus: *Computable Analysis*, Springer ISBN:3-540-66817-9 (2000).

[2] Brattka, Vasco: *Computable versions of the uniform boundedness theorem*, unpublished.

[3] Brattka, Vasco / Hertling, Peter: *Topological properties of real number representations*, Theoretical Computer Science, Volume 284, Issue 2 (July 2002).

[4] Priestley, Hilary A: *Introduction to Integration*, OUP ISBN:0-19-850123-4 (1997).

[5] Priestley, Hilary A: *Introduction to Complex Analysis*, OUP ISBN:0-19-853428-0 (1989).

[6] Edwards, Robert E: *Fourier Series A Modern Introduction* Volume 1 Springer-Verlag ISBN:0-387-90412-3 (1967).

[7] Plymble, L / Sanders, J.W: *Computability, Quadrature and the Fourier Transform* New South Wales Institute of Technology (1981).

[8] Pour-El, Marian B. / Richards J. Ian *Computability in Analysis and Physics* Springer-Verlag ISBN:0-387-50035-9 (1989).

[9] Roscoe, A.W *Notes on Domain Theory* Oxford University Computing Laboratory.