There is no algorithm to determine if the homotopy group is trivial or not for all compact spaces

Dayou Gao[1,5], Jiaxi Huang[2,6,*], Tianchang Liu[3,7], Tiantian Liu[4,8]

[1]Olive Tree International Academy, BFSU, Hangzhou, China
[2]Sun Yat-Sen University, Guangzhou, China
[3]Southeast University, Nanjing, China
[4]Shanghai YK Pao High school, Shanghai, China

[5]1722691147@qq.com
[6]huangjx237@mail2.sysu.edu.cn
[7]liutc@gmx.com
[8]liutiantian2006@gmail.com
[*]corresponding author

Abstract. Constructive mathematics involves numbers and objects that can be constructed and computed through algorithmic methods. In this paper we prove that there is no algorithm to determine if the homotopy group π_n is trivial or not for all constructive compact spaces. In the beginning we introduce constructive mathematics and topology theory. Then we introduce constructive compact spaces to describe the compact space in algorithmic way, and we calculate certain homotopy groups. Finally, we use an unextendible partially computable function to prove that there is no algorithm that can always determine the triviality of their homotopy groups by contradiction.

Keywords: compactness, constructive mathematics, homotopy group, algorithm.

1. Introduction

Differing from classical mathematics, constructive mathematics involves "constructing" the mathematical object. This can be regarded as a turning point for mathematics based on intuitionistic thinking [1]. In 1907, Brouwer began to critique classical mathematics and brought the idea of "intuitionism." In his idea, a mathematical object exists only if it can be constructed mentally and formally [1]. In 1936, amidst developments in computer machines done by Turing, Statistics of Repetitions, and *The Applications of Probability to Crypt*, concepts of algorithmic computability were formalized [1,2]. Now, two separate branches evolved from constructive mathematics in the works of A. A. Markov, E. Bishop, and N. A. Shanin [1,3]. Markov introduced certain elementary objects, forming constructive objects that are constructed according to definite rules. Whereas Bishop showed that much of mathematics can be done constructively and without using somewhat questionable principles [1]. The key distinction between Bishop's constructive mathematics and Markov–Shanin's lies in the latter's acceptance of the principle of constructive choice, which occasionally permits the use of construction arguments leading to contradictions [4].

For many years, topology has been one of the most thrilling and impactful areas of study in modern mathematics [5]. The first application was in 1736, when Leonhard Euler considered the Königsberg bridge problem; he introduced the idea of networks of vertices connected by edges, introduced the Latin phrase analysis situs, and motivated the development of topology [5]. In the 19th century, German mathematician Johann Listing, known for the first printed use of the term topology, sought to comprehend the topological properties of surface-like objects formed by combining basic shapes like polygons or polyhedra [5]. In 1851, German mathematician Bernhard Riemann examined surfaces connected to complex number theory and used combinatorial topology as a method for analysing functions [5]. Subsequent work by numerous mathematicians led up to the 1895 publication of Analysis Situs, where Poincaré laid the groundwork for applying algebraic concepts in combinatorial topology. He introduced the notions of "homology" and "homotopy" [5].

Today, research in algebraic topology has thrived and resolved numerous significant questions. For example, issues related to compactness, a property that extends closed and bounded subsets of n-dimensional Euclidean space, are crucial in topology [5].

Our work, combining algorithms with topology, is intended to show that there is no algorithm to determine the triviality of the homotopy group π_n for all compact constructive spaces.

1.1. Constructive Mathematics

1.1.1. Algorithm

An algorithm is considered to be a set of certain operations. Formally, an algorithm can be expressed as a Turing Machine, involving a set of rules represented by a list of strings from a given finite alphabet A, with legal input and output. Let the set of all algorithms be Alg. Because the algorithm can be represented by a finite list of words in alphabet, Alg is countable. Constructively we can get an injection from given algorithms to natural numbers.

Theorem 1.1.1 There is a constructive injection $\mathcal{A}: Alg \to \mathbb{N}$. [6]

Sketch of Proof. By saying the injection is constructive, we mean we can realize the transformation in finite steps. Every algorithm can be given as a string of letters in an alphabet. We identify the alphabet with unique natural numbers with a given length of digits, such as the ASCII code, and combine the digits of the number together to generate a long but finite sequence of digits. The resulting natural number is what we need. It is unique because we can uniquely extract the digits from the beginning to the end to reveal the characters of the algorithm.

1.1.2. Constructive Real Numbers

A constructive real number expressed as a string $\alpha\Diamond\beta$. In the string, α is an algorithm that generates a Cauchy sequence consisting of rational numbers, where β is an algorithm that generates a natural number sequence, satisfying the following property:

$$\forall n \in \mathbb{N}, \forall i, j \in \mathbb{N}, i, j > \beta(n), |\alpha(i) - \alpha(j)| < 2^{-n} \tag{1}$$

To be specific, $\beta(n)$ controls its convergence speed. $\beta(n)$ is also called the *convergence regulator*. The convergence regulator is considered to be *standard* if $\beta(n) = n$ for natural number n. [7]

1.1.3. Enumerable and decidable sets

Definition 1.1.2 A computable function is an algorithm with a natural number input to a natural number output. For a computable function u with given input n, if u never terminates with input n, we say $u(n)$ is undefined. If u terminates and output m, we say $u(n)$ is defined. In that case, we say $u(n) = m$ [7]

Definition 1.1.3 Let μ be a set of words in an alphabet A, we say μ is *decidable* if there is an algorithm u that applies to all words in A, such that:

1) If the input $n \in \mu$, the algorithm will output 1
2) if the input $n \notin \mu$, the algorithm will output 0. [7]

Definition 1.1.4 Let μ be a set of words in an alphabet A, we say μ is *enumerable* if it is related to an algorithm u over $\mathbb{N}$, such that:

1) If $u(n)$ terminates for $n \in \mathbb{N}$, $u(n) \in \mu$.
2) For all $a \in \mu$, there is a $n \in \mathbb{N}$ such that $u(n)$ is exactly a. [7]

Lemma 1.1.5 The domain of a computable function is enumerable. [8]

Proof. Suppose a computable function has the domain μ. We can construct an algorithm u as following: Run the function for an integer n, if the function terminates, output n instead of original output. Now μ is enumerable because u is the enumerating algorithm.

Decidable sets are enumerable, but the converse is not always true.

Theorem 1.1.6 There is a computable function that does not admit an everywhere extension.

Proof. Let us choose some programming language and arrange all the programs that compute every function into a computable natural number sequence $p_0, p_1, \cdots$ by **Theorem 1.1.1**. Set $U(i,x)$ to be equal to the output that the program of number i run with input x. The function U is regarded to be the computable universal function. The desired function can be defined by setting $d'(n) = d(n) + 1$, where d is diagonal, i.e., $d(n) = U(n,n)$. Indeed, any of its total computable extensions $\overline{d'}$ differ from d everywhere (if $d(n)$ is undefined, then $\overline{d'}(n) \neq d(n)$, since $\overline{d'}$ is a total function. Otherwise, $d'^{(n)} = d(n) + 1$ is defined and $\overline{d'}(n) = d'(n) \neq d(n)$;). However, $\overline{d'}$ is not computable because $U(i,n)$ includes all possible functions that are computable, giving a contradiction. [8]

There are everywhere-defined computable functions, such as the constant function which just output 0 immediately for every input. But according to the proof of **Theorem 1.1.6** we can make a computable function that cannot be defined everywhere.

Proposition 1.1.7 There is an enumerable but not decidable subset $F \subset \mathbb{N}$.

Proof. Consider a computable function f with natural arguments and values that has no total computable extension. Its domain F is what we expect. Indeed, F is enumerable (by one of the definitions of enumerability). If F were decidable, then the function

$$g(x) = \begin{cases} f(x), & x \in F \\ 0, & x \notin F \end{cases} \tag{2}$$

would be a total computable extension of f (to compute $g(x)$, we check if x belongs to F (we can do this since F is decidable), and we compute $f(x)$ if $x \in F$). [8]

1.2. Algebraic Topology

Definition 1.2.1 Let f and g be continuous functions from two topological spaces X and Y. A *homotopy* between f and g is defined as a function $H: X \times [0,1] \to \mathrm{Y}$ that is continuous, such that $H(x,0) = f(x)$ and $H(x,1) = g(x)$ for all $x \in X$. [4]

Definition 1.2.2 In general, a homotopy $f_t: X \times [0,1] \to Y$ is called a *homotopy relative to A* when its restriction to a subspace $A \subset X$ is independent of t. For short, f_t is a *homotopy rel A*. [9]

Definition 1.2.3 Let I^n be the product of n unit intervals $[0,1]$, i.e., the n dimensional unit cube. We can deduce that the boundary ∂I^n consists of points with at least one coordinate equal to 0 or 1. For a space X and $x_0 \in X$, consider the set of continuous maps $f: I^n \to \mathrm{X}$, where f are required to satisfy $f(\partial I^n) = x_0$. Functions f and g are considered equivalent if there is a homotopy f_t relative to ∂I^n between them. Let $\pi_n(X, x_0)$ to be the set of equivalence classes of these maps, where x_0 is said to be the base point.

For homotopy classes $[f]$and $[g]$ and $x = (x_1, x_2, \cdots, x_n) \in I^n$, define the multiplication $[f] \circ [g]$ to be the equivalent class $[h]$, where

$$h = \begin{cases} f(2tx_1, x_2, \cdots, x_n), & 0 \leq t \leq 1/2 \\ g\big((2t-1)x_1, x_2, \cdots, x_n\big), & 1/2 < t \leq 1 \end{cases} \tag{3}$$

We can check that the multiplication is well defined and $\pi_n(X, x_0)$ is a group under this multiplication.

The *homotopy group*, denoted as $\pi_n(X, x)$, can also be defined as the set of homotopy classes of maps with domain n-sphere S^n (the space of unit vectors in $\mathbb{R}^{n+1}$) to X which send a fixed point, denoted in the case as $x \in S^n$ (called the base point) to a fixed point in X.

These two definitions, disregarding of different expressions, are equivalent. [9]

Lemma 1.2.4 Every homotopy group of a closed interval in $\mathbb{R}$ is trivial. [9]

Proposition 1.2.5 For a product space $\prod_{i=1}^{n} X_i$ of a collection of path-connected spaces X_i there is an isomorphism $\pi_j(\prod_{i=1}^{n} X_i) \cong \prod_{i=1}^{n} \pi_j(X_i)$ for all j. [9]

Theorem 1.2.6 $\pi_n(S^n) = \mathbb{Z}$. [9]

Lemma 1.2.7 Let X be n dimensional closed cube with side length $d \in \mathbb{R}$, and Y be a n dimensional open cube having the same center as X and side length is $\acute{d} < d$. Let Y^c be the complementary of Y , then $\pi_n(X \cap Y^c) = \mathbb{Z}$.

Proof. By **Lemma 1.2.4** and **Proposition 1.2.5,** we have that $\pi_n(X \cap Y^c) = \pi_n([\acute{d}, d] \times S^n) = \pi_n(S^n) \times \pi_n([\acute{d}, d]) = \pi_n(S^n) = \mathbb{Z}$.

2. Preliminary

To find a function to determine whether a given path-connected compact space has trivial homotopy group or not, we firstly need to formally find a way to describe the compact space, by adopting the idea of ε-net.

Definition 2.1 A finite ε-net of a compact set A in a metric space (X, ρ) is a finite set $P = x_1, x_2, \cdots, x_m$ with $x_i \in X$, $i = 1,2, \cdots m$, such that for every $y \in A$ there exists a $x_i \in P$ and $\rho(x_i, y) < \varepsilon$.

To be precise, a ε-net gives the centers of balls with radii ε such that the balls give a finite cover of A [9].

Definition 2.2 A point $x = (p_1, p_2, \cdots, p_n) \in \mathbb{R}^n$ is a rational point if for all $k = 1, \cdots, n$, p_k is a rational number. [9]

The following definition comes from the standard definition of a compact space in a metric space that is complete.

Definition 2.3 A *constructive compact topological space* is an algorithm that generates a sequence of finite ε-nets $P_k: k = 1,2, \cdots$, for ε of the form $1/2^k, k \in N$, such that for all $x \in P_k$ for $k \geq 2$, there exists a point $x' \in P_{k-1}$, with $\rho(x, x') \leq 2^{-k}$.We better say the compact space is the closure of the union of these epsilon nets. [9]

The point $x \in P_k$ in the ambient space which in our case is a cube in the Euclidean space should be chosen rational. Both the unit cube and its boundary are constructive compact topological spaces. For the ε-nets of the cube we can take all the points with integer enumerator and the denominator that is the power of 2. For the boundary of the cube, we can do a similar construction, but it requires that one of the coordinates of the points of the ε-net is 0 or 1.

3. Theorem and Proof

Theorem 3.1 There is no algorithm to determine whether a given constructive compact space has trivial homotopy group π_{d-1} or not for $d \geq 2$.

Without loss of generality, we only consider the constructive compact space is path-connected. Take a computable function u that does not admit an everywhere defined extension. For every integer $n \in \mathbb{N}$, we intend to design an algorithm c_n generating a constructive compact space in $\mathbb{R}^d$. We design the algorithm as follows:

For input $n \in \mathbb{N}$, we change the algorithm u to count the step it currently runs for, where the step k refers to a natural number.

Let point $(x_1, x_2, x_3, \cdots)/c$ means $(x_1/c, x_2/c, x_3/c, \cdots)$.If $u(n)$ does not terminates at step k, set

$$P_k = \{a \in \mathbb{R}^d: a = (x_1, x_2, \cdots, x_d)/(3^k d); x_i = 0,1, \cdots, 3^k; i = 1,2, \cdots, d\} \tag{4}$$

If $u(n)$ terminates at step m, for every $k \geq m$, let

$$Q_k = \{a \in \mathbb{R}^d : a = (x_1, x_2, \cdots, x_d)/(3^k d); x_i = 0,1,\cdots,3^k; i = 1,2,\cdots,d\} \tag{5}$$

$$\begin{gathered} R_k = \{a \in \mathbb{R}^d : a = (x_1, x_2, \cdots, x_d)/(3^k d); \\ x_i = \frac{(3^m-1)3^{k-m}}{2} + 1, \frac{(3^m-1)3^{k-m}}{2} + 2, \cdots, \frac{(3^m+1)3^{k-m}}{2} - 1; i = 1,2,\cdots,d\} \end{gathered} \tag{6}$$

And set $P_k = Q_k \backslash R_k$. Basically P_k is the ε-net for the whole cube minus the $1/3^m$ side cube in the center of it.

In the end, let c_n be the algorithm generating the sequence $P_k : k = 1,2,\cdots$. It might be better to think of c_n as the finite algorithm that gives the sequence rather than the infinite sequence of ε-nets and this program is exactly the input into the computer program v below.

Lemma 3.2 For $n \in \mathbb{N}$, c_n satisfies the definition of a constructive compact space.

Proof. The proof of **Lemma 3.2** is clear. Note that the division by d in the definition is to make the points close enough.

Lemma 3.3 The constructive compact space of input n is related to the cube $\tilde{I}^d = (x_i) \in \mathbb{R}: 0 \leq x_i \leq 1/d$ in $\mathbb{R}^d$ if $u(n)$ is undefined. And if $u(n)$ is defined, the constructable compact space of input n refer to the cube with a hole $A^d_{H_m} = \{(x_i) \in \mathbb{R}^3 : 0 \leq x_i \leq 1/d\} \backslash \{(x_i) \in \mathbb{R}: (1/2 - 1/(2 \times 3^m))/d < x_i < (1/2 + (2 \times 3^m)/d\}$ for some $m \in \mathbb{N}$ in $\mathbb{R}^d$.

Proof. This can be verified by checking that P_k is a ε-net of them. If $u(n)$ does not terminates at step k, P_k is a ε-net of both A^d and $A^d_{H_m}$ for $m \geq k$.

If $u(n)$ terminates at step k, then P_k remain to be the ε-net of and $A^d_{H_k}$.

If $u(n)$ never terminates P_k is still the ε-net of A^d.

For undefined input n, $u(n)$ never terminates, so the compact space is A^d. For an input n, on which $u(n)$ terminates at some step m, the resulting compact space is $A^d_{H_m}$.

At the last step, we can prove **Theorem 3.1**.

Proof of theorem 3.1. Assume there exists an algorithm $v(n)$ to determine whether a given constructive compact space has trivial homotopy group π_{d-1}, such that v will output 1 if n refers to a constructive compact space has trivial homotopy and output 0 otherwise. The algorithm will also work on all the computer program c_n. By **Lemma 3.3**, if $\mathrm{u}(n)$ never terminates then the constructive compact space of input n is the cube A^d, otherwise it is the cube with a hole $A^d_{H_m}$. By **Lemma 1.2.4** and **Proposition 1.2.5,** and **Lemma 1.2.7** we know that $\pi_{d-1}(A^d) = 0$, and $\pi_{d-1}(A^d_{H_m}) = \mathbb{Z}$, which means that v will output 1 if the $u(n)$ never terminates and output 0 if the $u(n)$ ever terminates. This v is exactly the definition of the decidable set according to **Definition 1.1.3**, saying the domain of u is decidable. However, we know from **Proposition 1.1.7** that the domain of u is not decidable, giving a contradiction.

4. Conclusion

In this article we prove that there is no algorithm to determine whether a given constructive compact space has trivial homotopy group π_n or not for $n \geq 1$ by contradiction.

Using a similar method, we can also show that it is impossible to algorithmically decide if a general constructive compact topological space has $\pi_n = \mathbb{Z} + \mathbb{Z}$ or likely any other nice and reasonable group.

References

[1] Turing, A.M. (1937), On Computable Numbers, with an Application to the Entscheidungsproblem, Proceedings of the London Mathematical Society, 2, vol. 42, no. 1, pp. 230–65, doi:10.1112/plms/s2-42.1.230.

[2] Turing, A.M. (1938), On Computable Numbers, with an Application to the Entscheidungsproblem. A Correction, proceedings of the London Mathematical Society, 2. 43 (6): 544–6. doi:10.1112/plms/s2-43.6.544.

[3] Bishop, E., & Beeson, M. (2013), Foundations of constructive analysis. Ishi Press International.

[4] Laura, C., & Peter, S. (2005), From Sets and Types to Topology and Analysis: Towards practicable foundations for constructive mathematics. Oxford Science Publications.
[5] Carlson, S. (1999), Topology - History of topology, Encyclopedia Britannica, https://www.britannica.com/science/topology/History-of-topology
[6] Kushner, B. A. (1985), Lectures on constructive mathematical analysis. American Mathematical Society
[7] Bishop, E., & Douglas, B. (1985), Constructive Analysis. New York: Springer.
[8] Shen, A., and Vereshchagin N. K. (2003), Computable Functions. AMS Press.
[9] Hatcher, A. (2002), Algebraic Topology, Cambridge University Press, ed.1, pp.364, doi:10.1017/S0013091503214620.

Appendix

$\mathbb{N}, \mathbb{R}, \mathbb{Q}, \mathbb{Z}$	Natural numbers, Real numbers, Rational numbers, Integer numbers
Alg	Set of algorithms
$\mathcal{A}$	The constructive injection $Alg \to \mathbb{N}$
$\alpha \Diamond \beta$	Constructive real number
$(p_1, p_2, \cdots, p_n)$	Point in $\mathbb{R}^n$
$\cap, \cup, \setminus, c$	Intersection, Union, Subtraction, Complementary
$\|x\|$	Euclidian norm $\sqrt{(p_1^2 + p_2^2 + \cdots + p_n^2)}$ for $x = (p_1, p_2, \cdots, p_n)$
I	Interval [0,1]
I^n	Product of n interval [0,1]
π_n	Homotopy group
S^n	$\{x \in \mathbb{R}^{n+1} : \|x\| = 1\}$