A class of numbers which intuitively seem to be all of $\mathbb{R}$ but aren't!

Ashish Kujur
ashish.kujur21@iisertvm.ac.in

Joel Sleeba
sleeba21@iisertvm.ac.in

April 2022

1 Introduction

For us humans, the rationals are a familiar territory despite them being slightly more involved than the integers. Given enough supply of papers and pens, a **rational** number r and a natural number n to an immortal[1] human, he can compute the nth digit of r in a finite amount of time by using the division algorithm for $\mathbb{Z}$. Before we proceed any further, let's agree on the convention that whenever we want someone to compute the nth digit of a number, we want him to use the recurring 0 form instead of the recurring 9 form. For instance, $1 = 1.000\ldots = 0.999\ldots$ but due to our convention, the second digit of 1 is 0 and not 9. Let's look at another example: the first digit of $\frac{1}{9} = 0.\overline{1}$ is 0 and nth digit is 1 for any $n > 1$.

In a first course of real analysis, it is often demonstrated that every real number can be approximated by a rational number to arbitrary accuracy. Mathematically speaking, for every $\varepsilon > 0$ and every real number a, there is a rational r such that $|a - r| \leq \varepsilon$. So, let us ask a question: Given enough supply of papers and pens, a **real** number r and a natural number n to an immortal human, can he compute and produce the nth digit of r by using some algorithm?

The answer to this question may seem yes and here's how any layperson would argue to prove that it can be done: Given any real number a, the

[1] We grant immortatility to that human because we don't want him to pass away before he does our job!

density theorem gives us a rational number r such that $|r-a| < \frac{1}{10^{n+1}}$ and we can compute the nth digit of r because r is rational. Since the the distance $|r-a| < \frac{1}{10^{n+1}}$, we know the nth digit of a is the nth digit of r. However, this argument is wrong and we encourage the reader to they figure out why (Correspondences are welcome!).

2 Getting a little more Mathematical!

Let's make a definition at this point:

Definition: Computable Number

*We will call a positive real number a to be **computable** if there is an algorithm or a mechanical procedure (that can be performed by a programming language, an immortal human, or whatever) which takes a natural number n as an input and returns the nth digit of a as an output.*

All rational numbers are computable because the division algorithm is an algorithm which allows us to compute the nth digit of any given rational number. This leads us to ask this question: are all the real numbers computable or are the only computable numbers rational?

In 2014, Brady Haran from Numberphile, in this video printed out 1 million digits of π on the event of his Numberphile channel surpassing 1 million subscribers. This gives us a good reason to believe that π is computable and indeed this is the case! Several other transcendental numbers like e are computable as well.

3 Meeting a transcendental number which is computable.

We sketch a proof of why π is computable. The proof is simply an algorithm as the definition demands. To show this, we consider the Hutton's series[2] for

[2]We choose a series which does converge to π albeit not in an alternating fashion to guarantee faster convergence. See here for some more.

π:

$$\pi = \sum_{n=0}^{\infty} \frac{(n!2^n)^2}{(2n+1)!} \left(\frac{12}{5} \left(\frac{1}{10} \right)^n + \frac{14}{25} \left(\frac{1}{50} \right)^n \right) = \sum_{n=0}^{\infty} a_n \text{ (say)}$$

Let $S_k = \sum_{n=0}^{k} a_n$. It can be shown that $S_k < \pi < S_k + \frac{1}{10^k}$ for each $k \geq 0$. Let the input to the algorithm be n and our output will be the nth digit of π. Here's the algorithm:

1. Find an $N > n$ such that $S_N = a_0.a_1a_2 \ldots a_n \ldots a_N$ and not all of $a_{n+1}, a_{n+2}, \ldots, a_N$[3] are 9. Note that if it happened that for all $N > n$, $a_{n+k} = 9$ for all $k \in \mathbb{N}$ then π would be rational which it is certainly not!

2. And now we return a_n as the required output!

To see that nth decimal place of π is indeed a_n, consider the following: Select an m such that $a_m \neq 9$. Then by property of the series, we have

$$\begin{aligned} S_N < \pi &< S_N + \frac{1}{10^N} \\ &\leq S_N + \frac{1}{10^m} \end{aligned}$$

Hence, we have

$$a_0.a_1 \ldots a_n \ldots a_m \ldots < \pi < a_0.a_1 \ldots a_n \ldots (a_m + 1) \ldots$$

And so the nth digit of π is a_n!

With a "similar" algorithm, it can be shown that $e = \sum_{k=0}^{\infty} \frac{1}{k!}$ is computable.

4 Pièce De Résistance

So, we've managed to beat around a bush far enough to avoid the question at the question at hand:

> What is this class of numbers which intuitively seem to be all of $\mathbb{R}$ but aren't?

[3]Note that this can be done because S_n will rational regardless of the choice of n.

(Drumroll please!) It's the set of computable numbers $\mathcal{C}$! In the third paragraph of Introduction (Section 1), we argued why it may look like $\mathbb{R}$ but it's not! In fact, we're going to show that $\mathcal{C}$ is countable. So, most of the numbers in the wilderness are in fact uncomputable!

To show this, take your favourite programming language, [4] call it $\mathcal{L}$. Take $\mathcal{L}$ to be Python, if you prefer. Let's count how many programs we can possibly write in $\mathcal{L}$. Obviously, the number of symbols that we use to write in any programming language is finite and any program contains finitely many lines of instructions (lest it may not terminate!). So, the set of programs in $\mathcal{L}$ can be countable.

For each computable number x, there is some program P_x which takes a single natural number n as an input in the language $\mathcal{L}$ and computes its nth digit and outputs it. Since the set of programs in $\mathcal{L}$ is countable, $\mathcal{C}$ is computable.

You may now ask this question since our proof depended on the choice of our programming language $\mathcal{L}$:

> Are there procedures that can be done by pen-pencil methods but not by some programming languages and conversely?

There's no definitive answer to this question. Any mathematical model of computation[5] that we've come up with they coincide. And in fact, most computability theorists, logicians, computer scientists take the Church Turing thesis for granted which says that any procedure that can be "done on a computer" can be achieved by "pen pencil methods" and conversely. We are deliberately being vague as we are very ill-equipped to state this. We do not go any deeper into this.

The set of computable numbers is countable (the smallest kind of infinite), however, can you name an uncomputable number? Try but remember that if you give a good enough description of that number, we can always write a program which produces its nth digit. So, is it possible to find an uncomputable number? The answer is yes but not in an explicit way. We encourage the readers to construct an uncomputable number using the diagonal argument.

[4] There's a catch here: we need that programming language to be Turing complete. Let's just avoid the technicalities of this for this article as most of our favourite programming languages tend to be Turing Complete unless they are severely crippled.

[5] Turing Machines, Church's λ-calculus, Godel's partial recursive functions, to name a few.

5 Final Words

It was Alan Turing who introduced the notion of computable numbers in 1936 along with a negative response to Entscheidungsproblem (which was also independently resolved by Alonzo Church in 1936). It was shown in 1954 that the set of computable numbers $\mathcal{C}$ is a field which contain the algebraic real numbers. We showed that π is computable thus making $\mathcal{C}$ strictly larger the set of algebraic numbers (see figure 1).

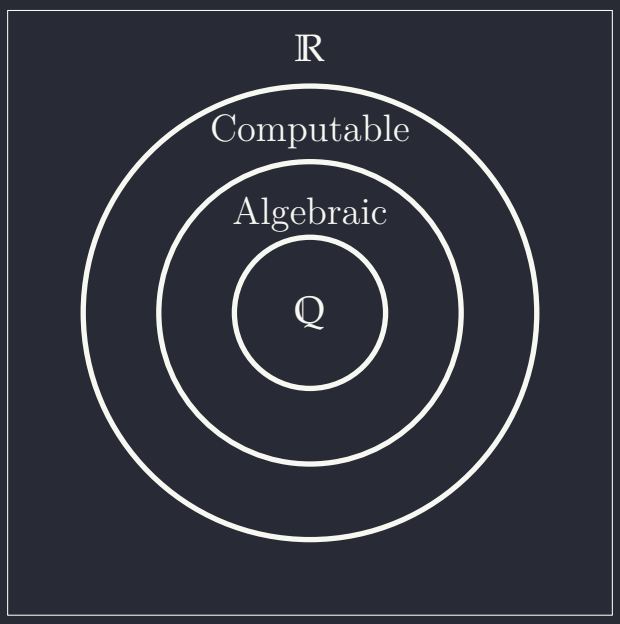

Figure 1: Where $\mathcal{C}$ lies...