Sorting Real Numbers, Constructively

Jules Jacobs

January 22, 2021

This note was inspired by a talk of Mikkel Kragh Mathiesen at OPLSS.

1 Introduction

We can represent a real number $r \in \mathbb{R}$ as Cauchy sequence $r_i \in \mathbb{Q}$ of rational numbers that converges to the real number we want to describe. We can represent an infinite sequence r_i as a computer program that calculates the function $r : \mathbb{N} \to \mathbb{Q}$, with which we can get arbitrarily precise approximations to r.

In order get a guaranteed ϵ-approximation, we must also have a constructive witness to the Cauchy property, that is, a way to determine how far into the sequence we have to look: given an $\epsilon > 0$ we must be able to determine an $n \in \mathbb{N}$ such that $r_{n+1}, r_{n+2}, \cdots \in r_n \pm \epsilon$, where $r_n \pm \epsilon = \{x : r_n - \epsilon \leq x \leq r_n + \epsilon\}$ is an interval with radius ϵ around r_n. One can do many operations on these real numbers:

- Arithmetic with $(r+s)_i = r_i + s_i$.
- Calculate $x = \sqrt{2}$ with $x_0 = 1$ and $x_{n+1} = \frac{x_n + 2/x_n}{2}$.
- Calculate transcendental functions $y = \sin(x)$ by taking $y_n = f_n(x_n)$ where f_n is the n-term Taylor series of sin.

2 What Constructivists Cannot Do

One thing you cannot do constructively is define a function that compares real numbers for equality. There are many different representations of the number zero: $x_n = 0$ and $y_n = 1/n$ because they both converge to zero. We say that two numbers x, y are equal if their difference $x - y$ is zero. It is not possible to constructively determine whether two numbers are equal, because you'd have to look infinitely far into the sequence. For the same reason, it's not possible to determine whether $x < y$ or $x \leq y$ for arbitrary real numbers x, y. This remains true even if we had access to the source code of the computer programs for calculating the sequences x, y, due to Gödel's incompleteness.

It would therefore seem to be impossible to sort an array of real numbers $[x_1, x_2, \ldots, x_n]$. The way we usually sort arrays is to determine a permutation that makes the array sorted, but in order to determine that permutation we'd have to compare the x_i with each other.

In general, it is only possible to calculate *continuous* functions of real numbers. The function $f(x) = 1$ if $x < 2$ and $f(x) = 0$ otherwise is not continuous: changing x by a little bit may change $f(x)$ by a discrete step. The same holds for the permutation: changing one of the x_i by a little bit may change the permutation required to put the array in sorted order, which is a discrete step change.

The situation is even worse. We usually say that a sort function is correct if its output is sorted, and if the output is a permutation of the input. Constructively, the latter entails a method to determine that permutation, which is impossible.

3 A Glimmer of Hope

Let sort : $\mathbb{R}^n \to \mathbb{R}^n$ be a sort function, presumably given to us by a non-constructivist. The sort function itself *is* continuous! To intuitively see this, let $[\dots, a, b, \dots]$ be a sorted array where each element is strictly smaller than the rest. If we slowly increase the value of a while keeping the other values fixed, there will be some point when $a > b$ and the sorted array becomes $[\dots, b, a, \dots]$. This may look like a non-continuous change, but it's not. To see this, let us consider what that change looks like:

$$
\begin{gathered}
[\dots, 1.7, 2.0, \dots] \\
[\dots, 1.8, 2.0, \dots] \\
[\dots, 1.9, 2.0, \dots] \\
[\dots, 2.0, 2.0, \dots] \\
[\dots, 2.0, 2.1, \dots] \\
[\dots, 2.0, 2.2, \dots] \\
[\dots, 2.0, 2.3, \dots]
\end{gathered}
$$

It looks like we first increased a, and when a became equal to b, we started increasing b. This is a continuous change: a gradual change in the input resulted in a gradual change in the output. We did switch *which* entry we were increasing, but two adjacent arrays in the preceding list only differ by a small amount. We therefore have some hope: considerations of continuity do not rule out a constructive sort function.

4 Sorting Reals

Similar to the preceding section, we can view an array $[a, b, \dots, z]$ of real numbers as a matrix of rational numbers, where the i-th row is an array of the i-th elements of the Cauchy sequences:

$$
\begin{matrix}
[a_0, & b_0, & \dots, & z_0] \\
[a_1, & b_1, & \dots, & z_1] \\
[a_2, & b_1, & \dots, & z_2] \\
\vdots & \vdots & \cdots & \vdots
\end{matrix}
$$

The output of our sorting algorithm must again be such a matrix. Since each row is an array of rational numbers, we can simply sort each row:

$$
\begin{matrix}
\mathsf{sort} & [a_0, & b_0, & \dots, & z_0] \\
\mathsf{sort} & [a_1, & b_1, & \dots, & z_1] \\
\mathsf{sort} & [a_2, & b_1, & \dots, & z_2] \\
\vdots & \vdots & \vdots & \cdots & \vdots
\end{matrix}
$$

Each column of the sorted matrix is a sequence of rational numbers, so it is potentially a real number. To prove that this method is correct, we must show:

1. Each column is a Cauchy sequence
2. The output real numbers are sorted
3. The output real numbers are a permutation of the input

The second property clearly holds: if $a_i \leq b_i$ for all i, then also $a \leq b$ in the sense of real numbers. I hope that the first property seems plausible after the preceding section, but the third property may seem obviously false: obviously the columns of the sorted matrix are not a permutation of the original columns.

This is true, but remember that equality of real numbers a, b is *not* equality of $a_i = b_i$ for all i. Since we have already seen that we cannot constructively establish property 3, we reason classically.

First, let us suppose that the real numbers $a, b, \ldots, z$ are all distinct. This means that there is some index i beyond which the Cauchy sequences of $a, b, \ldots, z$ all separate: the relative order of the rational numbers in their Cauchy sequence no longer changes. The permutation that sort will apply stabilizes after row i. Since only the tail of the sequences matters, the output of sort will be a permutation of the input in this case.

The interesting case is when some of the real numbers are equal. If $a = b$ then the relative order of the rationals in their Cauchy sequence may never stabilize. Consider $a_i = 0$ and $b_i = (-1)^i/i$. At even i we have $a_i < b_i$ but at odd i we have $b_i < a_i$. Thus, the permutation that sort applies will never stabilize.

If we group the input array in groups of real numbers that are equal, then the relative order of the groups *will* stabilize because their Cauchy sequences will separate, but the order within each group may not. The key point is that *this does not matter*: since the real numbers in each group are all equal, shuffling their Cauchy sequences around doesn't change a thing. If we have two real numbers $a = b$, then we can create a new real number c by picking c_i to be a_i or b_i arbitrarily, and we'll still have $a = b = c$.

The conclusion is that we can define the sort function constructively, but proving that its output is a permutation of the input can only be done classically.